Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Detekce polarity textu s využitím mezijazyčné transformace

Plzeň 2021 Jakub Šmíd

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd

Akademický rok: 2020/2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Jakub ŠMÍD**
Osobní číslo:	**A19B0675P**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Informatika**
Téma práce:	**Detekce polarity textu s využitím mezijazyčné transformace**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Seznamte se se základními metodami strojového učení pro klasifikaci textu a s vybranými metodami pro reprezentaci významu textu, včetně možností jejich transformace mezi jazyky.
2. Vyberte vhodný model pro klasifikaci textu a vytvořte program umožňující detekci polarity českého textu.
3. Proveďte experimenty na datech dodaných vedoucím práce s použitím mezijazyčné transformace mezi angličtinou a češtinou.
4. Výsledky experimentů porovnejte a zhodnoťte vliv mezijazyčné transformace na úspěšnost klasifikace.

Rozsah bakalářské práce: **doporuč. 30 s. původního textu**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná**

Seznam doporučené literatury:

Dodá vedoucí bakalářské práce.

Vedoucí bakalářské práce: **Ing. Pavel Přibáň**
Nové technologie pro informační společnost

Datum zadání bakalářské práce: **5. října 2020**
Termín odevzdání bakalářské práce: **6. května 2021**

L.S.

Doc. Dr. Ing. Vlasta Radová
děkanka

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 26. října 2020

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 5. května 2021

Jakub Šmíd

Poděkování

Rád bych poděkoval Ing. Pavlu Přibáňovi za příkladné vedení bakalářské práce, odborné rady, cenné připomínky, ochotu a trpělivost. Také děkuji za výpočetní zdroje poskytnuté projektem *e-Infrastruktura CZ* (e-INFRA LM2018140) zajištěným v rámci programu Velké infrastruktury pro výzkum, vývoj a inovace.

Abstract

This bachelor's thesis focuses on text polarity detection using cross-lingual transformations. Cross-lingual transformations are among the methods that allow the transfer of knowledge between languages. Specifically, in this work, the English annotated data are used for polarity detection in Czech text. Within the thesis, two neural network models – LSTM and CNN – are proposed and implemented in combination with *fastText* word embeddings. The models are then trained on English data and evaluated on Czech data using linear cross-lingual transformations. The results of these experiments are compared with models that have been trained and evaluated only on Czech data. The comparison shows that with a sufficient amount of exclusively English data, it is possible to achieve very good results, which are only 5 to 6% worse compared to models trained only on Czech data.

Abstrakt

Tato bakalářská práce se zabývá detekcí polarity textu s využitím mezijazyčných transformací. Mezijazyčné transformace patří mezi metody, které umožňují přenos znalostí mezi jazyky. Konkrétně v této práci jsou použita anglická anotovaná data pro detekci polarity v českém textu. V rámci práce jsou navrženy a implementovány dva modely neuronových sítí – LSTM a CNN – v kombinaci se slovními vektory *fastText*. Modely jsou následně s využitím lineárních mezijazyčných transformací natrénovány na anglických datech a vyhodnoceny na českých datech. Výsledky těchto experimentů jsou porovnány s modely, které byly trénovány i vyhodnoceny pouze na češtině. Srovnání ukazuje, že s dostatečným množstvím výhradně anglických dat lze dosáhnout velmi dobrých výsledků, které jsou jen o 5 až 6 % horší v porovnání s modely trénovanými jen na českých datech.

Obsah

1 Úvod

Zpracování přirozeného jazyka (NLP, z angl. *natural language processing*) je obor na pomezí informatiky, matematiky a lingvistiky, který se zabývá analýzou a zpracováním nestrukturovaného textu. Začal se rozvíjet v 50. letech 20. století. Se zpracováním přirozeného jazyka je možné se setkat v mnoha oblastech, např. ve vědě, ekonomice, ale i v běžném životě například ve formě internetového vyhledávače a překladače. Úlohy patřící do zpracování přirozeného jazyka jsou např. strojový překlad, analýza sentimentu, vyhledávání informací či sumarizace textu [8].

Tato práce se zabývá **detekcí polarity textu s využitím mezijazyčných transformací pro její řešení**. Detekce polarity je úloha spadající pod analýzu sentimentu (dalšími úlohami jsou například detekce emocí) [38]. Cílem je klasifikovat vstupní text podle jeho polarity – pozitivní, negativní, případně neutrální. Toho lze využít např. u uživatelských recenzí obchodních značek (firem), služeb či produktů, které je možné hodnotit na internetu.

Většina označených (anotovaných) dat pro úlohy zpracování přirozeného jazyka je obvykle dostupná pro anglický jazyk, pro ostatní jazyky je dat málo či vůbec žádná, jelikož jejich získání je časově náročné, drahé a obtížné. Z tohoto důvodu se hledají metody, které by dovolily použití dat z jazyků s velkým množstvím dat (angl. *resource-rich languages*) – typicky angličtina – v jazycích, kde je dat méně (angl. *low-resource languages*). Jednou z možností jsou **mezijazyčné transformace**.

Cílem této práce je vytvořit program pro detekci polarity českého textu a pomocí experimentů ověřit možnosti využití dat z jiného jazyka (konkrétně angličtiny) s použitím mezijazyčných transformací a zhodnotit jejich vliv na úspěšnost detekce polarity.

1.1 Úlohy zpracování přirozeného jazyka

Zpracování přirozeného jazyka je široká oblast; mezi nejznámější úlohy patří **detekce polarity**. Detekce polarity je běžně využívána různými společnostmi, což jim umožňuje porozumět názoru zákazníků o jejich produktech, službách či o společnosti jako takové. To je pro každou společnost velice cenné. Příkladem recenze televizoru, kterou lze označit za pozitivní, je *„Skvělý obraz! Sledovat filmy je jedna báseň.“*, naopak *„Televizor strašný. O nepřehledném ovladači ani nemluvím...“* je recenze negativní. Dalším vyu-

žitím je doporučovací systém (angl. *recommendation system*) [37], který uživatelům přizpůsobuje či doporučuje jakýkoli obsah (filmy, hudbu, reklamy, produkty apod.). Doporučení jsou založena na předchozí interakci uživatele se systémem, například ve formě komentářů, recenzí či hodnocení obsahu.

Mezi další úlohy patří **strojový překlad** (angl. *machine translation*). Cílem je překlad textu z jednoho jazyka do druhého [8]. Typickým příkladem využití strojového překladu jsou nástroje Překladač Google a Překladač Bing.

Dalšími úlohami zpracování přirozeného jazyka jsou **extrakce informací**, **filtrování spamů**, **kontrola překlepů**, **virtuální asistenti** (např. *Apple Siri*, *Cortana* a *Google Assistant*), **klasifikace dokumentů**, **vytváření jazykových modelů**, **rozpoznávání slovních druhů** (POS, z angl. *part-of-speech tagging*), **sumarizace textu**, **rozpoznávání pojmenovaných entit** (NER, z angl. *named entity recognition*) a **analýza sentimentu** [30].

2 Předzpracování a reprezentace textu

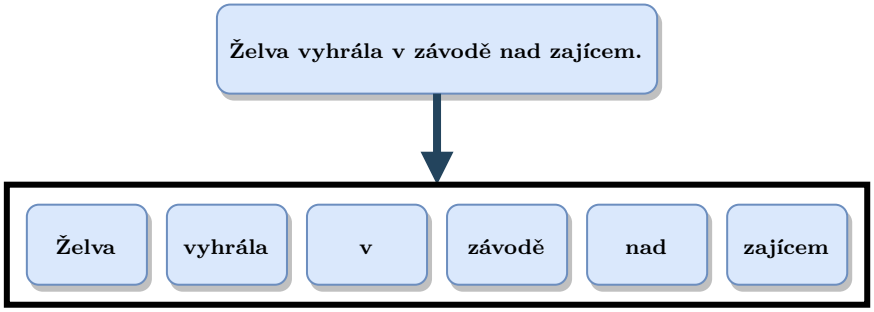
Před řešením většiny úloh NLP je třeba text patřičným způsobem předzpracovat a připravit do podoby použitelné pro metody strojového učení.

2.1 Předzpracování textu

Dřívější přístupy pro řešení úloh zpracování přirozeného jazyka požadovaly pokročilejší předzpracování vstupních dat, pro současné moderní přístupy již předzpracování téměř či vůbec není potřeba [8]. V této sekci jsou popsány některé často používané techniky předzpracování textu.

2.1.1 Tokenizace

Tokenizace je rozdělení textu na jednotlivé **tokeny** (viz obrázek 2.1). V procesu tokenizace jsou často z textu odstraněny určité znaky, např. interpunkce [41].



Obrázek 2.1: Příklad rozdělení věty na jednotlivé tokeny (tokenizace).

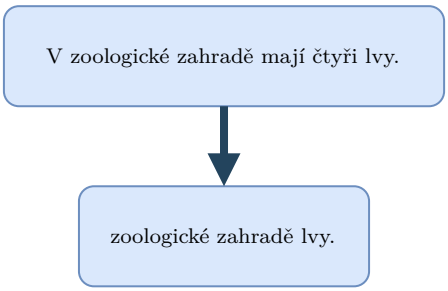
2.1.2 Stematizace a lematizace

Stematizace (angl. *stemming*) je převedení slova na tvar jeho kmene či kořene odstraněním předpon a koncovek [41]. Například pro slova *„prase“*, *„prasata“*, *„prasetem“*, *„praseti“* může být výstupem stematizace *„pras“*.

Lematizace převede slova na základní tvar zvaný lemma [41]. Například pro slova *„prase“*, *„prasata“*, *„prasetem“*, *„praseti“* je výsledkem lematizace základní tvar *„prase“*.

2.1.3 Odstranění stopslov

Stopslova (angl. *stop words*) jsou slova, která do věty nepřidávají významovou informaci (mají většinou pouze syntaktický význam), a proto mohou být vypuštěna. Tím je také redukována velikost slovníku [41]. Nevýznamová slova v českém jazyce jsou např. předložky, spojky, zájmena či číslovky, v angličtině např. členy *„the“* a *„a“* (příklad odstranění viz obrázek 2.2).



Obrázek 2.2: Příklad odstranění stopslov v českém jazyce.

2.1.4 Další metody předzpracování textu

Lowercasing je převedení všech písmen na jejich minuskulní podobu (převedení na malá písmena) [41]. Důvodem jsou např. velká písmena na začátku vět či chtěné zdůraznění slova použitím všech velkých písmen – ve většině případů je vhodné považovat např. slova *„skvělý“*, *„Skvělý“* a *„SKVĚLÝ“* za identická.

Obdobou je ***truecasing***, kdy je ve slově správně ponecháno velké písmeno tam, kde ve skutečnosti být má (např. u jmen) [41].

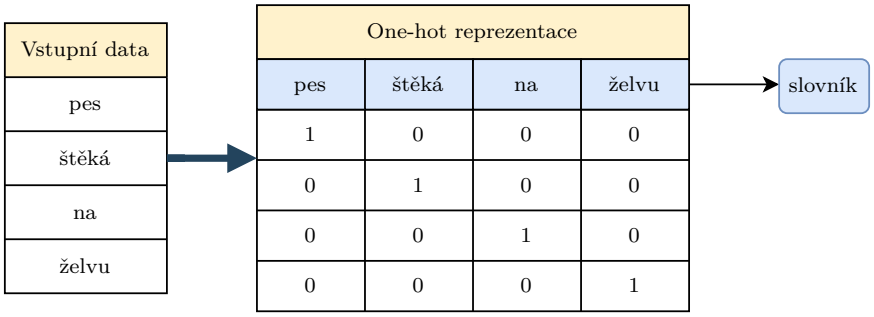
Posledním zmíněnou metodou předzpracování textu je **odstranění diakritiky**. Tuto metodu je vhodné použít, pokud se ve významném počtu vzorků vyskytuje text bez diakritiky tam, kde by se měla vyskytovat (např. uživatelé sociálních sítí často píší bez diakritiky) [41]. V takovém případě může být žádoucí považovat např. slova *„já“* a *„ja“* za shodná.

2.2 Reprezentace textu

Počítače jsou stroje pracující s čísly, slovům nerozumí. Proto je nutné převést slova (vstupní data) na číselné hodnoty (vektory). Tento proces se nazývá **vektorizace textu** – vstupní text je převeden na vektor čísel, se kterým již algoritmy strojového učení dokáží pracovat [8].

2.2.1 One-hot

One-hot vektor je vektor o velikosti rovné počtu všech různých vstupních hodnot (v NLP obvykle o velikosti slovníku). Slovo je reprezentováno tímto vektorem s pouze jednou nenulovou hodnotou na pozici odpovídající danému slovu ve slovníku (příklad viz obrázek 2.3) [30].



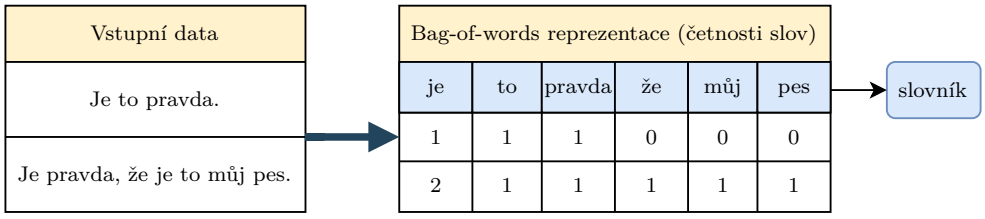
Obrázek 2.3: Příklad *one-hot* reprezentace.

2.2.2 Bag-of-words

Bag-of-words (BOW) je další základní metodou vektorizace textu. Model popisuje výskyt slov v daném textu [30]. Nejprve je vytvořen slovník všech slov z trénovacího korpusu. Dále se vytváří vektor pro každý dokument, který má nenulové hodnoty na pozicích, jež odpovídají indexům všech slov tohoto dokumentu ve slovníku. Tyto hodnoty se vyjadřují více způsoby.

Vyjádření četnosti

Na každé pozici vektoru je číslo udávající četnost daného slova v dokumentu (viz obrázek 2.4).



Obrázek 2.4: Příklad *bag-of-words* reprezentace vyjádřením četnosti.

Binární reprezentace

Na odpovídající pozici ve vektoru je hodnota 1, jestliže dokument obsahuje dané slovo, jinak 0. Alternativně lze 1 a 0 zaměnit za hodnoty *True* a *False*.

TF-IDF

TF-IDF (z angl. *term frequency – inverse document frequency*) je metoda přiřazující každému slovu váhu, která reprezentuje důležitost daného slova v rámci dokumentu [54].

TF (z angl. *term frequency*) vyjadřuje četnost slova v dokumentu a lze spočítat ze vztahu 2.1 [30], kde $n_{t,d}$ je počet výskytů slova t v dokumentu d.

$$tf_{t,d} = \log_{10}(n_{t,d} + 1) \tag{2.1}$$

IDF (z angl. *inverse document frequency*) reprezentuje důležitost slova v celém korpusu. Lze předpokládat, že často vyskytující se slova mají malou důležitost (např. sloveso *„být“* se pravděpodobně objeví často ve všech dokumentech, ale pro klasifikaci má malý význam). Důležitost slova idf_t se spočte ze vztahu 2.2 [30], kde $|D|$ udává počet všech dokumentů a df_t počet dokumentů, ve kterých se slovo t vyskytuje.

$$idf_t = \log_{10} \frac{|D|}{df_t} \tag{2.2}$$

TF-IDF váha slova t v dokumentu d vznikne vynásobením předchozích složek (viz vzorec 2.3). Další varianty výpočtu TF a IDF jsou zmíněny v [41].

$$tf\text{-}idf_{t,d} = tf_{t,d} \times idf_t \tag{2.3}$$

2.2.3 Sémantická reprezentace slov

Sémantické modely vychází z *distribuční hypotézy* [20, 24], která tvrdí, že slova objevující se v podobných kontextech mívají podobné významy. Základní myšlenkou tedy je, že význam slova je dán slovy často se vyskytujícími v jeho okolí. Sémantické modely na základě těchto tvrzení reprezentují každé slovo vektorem tvořeným obecně nenulovými reálnými čísly. Tyto slovní vektory se dají rozdělit na předtrénované a kontextualizované [25, 26].

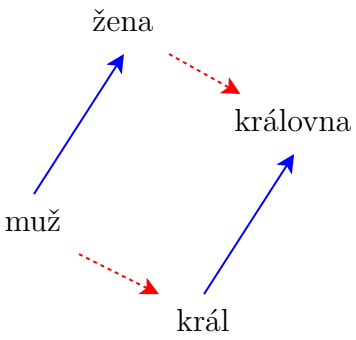
Podobnost a slovní analogie

Podobnost mezi slovy (jejich slovními vektory) je často měřena pomocí **kosinové podobnosti** (angl. *cosine similarity*), která se rovná kosinu úhlu α svíraného vektory $\mathbf{v}$ a $\mathbf{w}$ s dimenzí d a lze spočítat jako skalární součin těchto vektorů vydělený součinem jejich velikostí (viz rovnice 2.4) [30].

$$\cos(\alpha) = \cos(\mathbf{v}, \mathbf{w}) = \frac{\mathbf{v} \cdot \mathbf{w}}{\|\mathbf{v}\| \cdot \|\mathbf{w}\|} = \frac{\sum_{i=1}^{d} v_i w_i}{\sqrt{\sum_{i=1}^{d} v_i^2} \sqrt{\sum_{i=1}^{d} w_i^2}} \tag{2.4}$$

Pokud je úhel mezi vektory 0°, pak se kosinová podobnost rovná 1 a slova mají stejný či velmi podobný význam. Jsou-li však vektory na sebe kolmé (svírají úhel 90°), potom je kosinová podobnost rovna 0 a slova si nejsou sémanticky podobná.

Mezi známé vlastnosti slovních vektorů patří slovní analogie. Například dotaz vektor(*„král“*) − vektor(*„muž“*) + vektor(*„žena“*) vrátí velmi pravděpodobně jako odpověď vektor, který je nejblíže vektorové reprezentaci slova *„královna“*. Je to dáno lineární závislostí mezi vektory slov, kdy pokud se přičte k vektoru slova vektor reprezentující určitou sémantickou či morfologickou vlastnost, posune se k těsné blízkosti vektoru slova, které se liší od původního právě danou vlastností, jak lze vidět na obrázku 2.5 [48].



Obrázek 2.5: Vektorová reprezentace slov. Modré šipky představují vlastnost změny pohlaví z mužského na ženské, červené přerušované pak získání panovnické hodnosti.

Předtrénované slovní vektory

Předtrénované (též statické) slovní vektory (angl. *pre-trained word embeddings*) vytváří vektor pro každé slovo. Nevýhodou tohoto modelu je, že pro stejné slovo poskytuje vždy stejný vektor bez ohledu na konkrétní význam, který může být dán dalšími slovy ve větě. Například slovo *„los“* vyskytující se v kontextu *„los evropský“* a *„stírací los“* má vždy odlišný význam, ale je vyjádřeno jedním statickým vektorem. Statické slovní vektory mapují každé slovo w_i ze slovníku V na vektor $\mathbf{h}_i \in \mathbb{R}^d$ (viz rovnice 2.5) [25, 26].

$$f_V : w_i \rightarrow \mathbf{h}_i \tag{2.5}$$

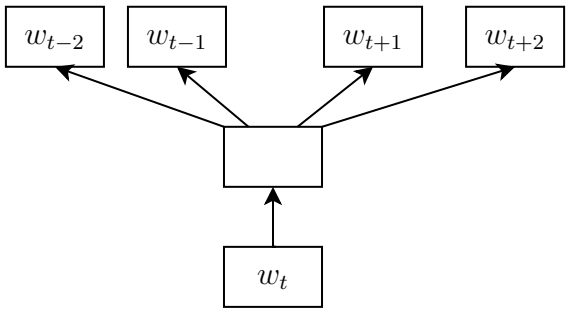
Word2vec *Word2vec* [45] je efektivní metoda pro vytváření slovních vektorů, která pro jejich tvorbu používá neuronovou síť. Myšlenkou tohoto modelu je projít každou pozici t v textu danou centrálním slovem c a sousedními (kontextovými) slovy o. Každé slovo w má dva vektory, $\mathbf{u}_w$ když w je

kontextové slovo a $\mathbf{v}_w$ když w je centrální slovo. Tyto vektory jsou uložené v maticích $\mathbf{U}$ a $\mathbf{V}$ o rozměrech $|V| \times d$, kde $|V|$ je velikost slovníku a d dimenze vektorů. Obě zmíněné matice jsou na začátku trénovacího procesu náhodně inicializovány a tvoří parametry $\boldsymbol{\Theta}$ tohoto modelu. Cílem je maximalizovat pravděpodobnost kontextového slova o za podmínky centrálního slova c (či naopak) vypočtenou použitím funkce *softmax* dle vztahu 2.6 [47], který lze použít i v dalších rovnicích se záměnou značení. Text je obvykle zpracován ve více iteracích. Výsledné slovní vektory lze získat např. zprůměrováním vektorů z obou matic [30]. *Word2vec* má dvě varianty – *Skip-gram* a CBOW.

$$P(o|c) = \frac{\exp(\mathbf{u}_o^\intercal \mathbf{v}_c)}{\sum_{w \in V} \exp(\mathbf{u}_w^\intercal \mathbf{v}_c)} \tag{2.6}$$

Skip-gram model předpovídá pro každou pozici $t = 1, \ldots, T$ kontextová slova v rozsahu velikosti m od daného centrálního slova w_t (viz obrázek 2.6). Například pro slovo *„džus“* předpoví s vysokou pravděpodobností kontextová slova *„snídani“*, *„pomerančový“* nebo *„chléb“*. Model je optimalizován minimalizací funkce $J(\boldsymbol{\Theta})$ ve tvaru 2.7 [47].

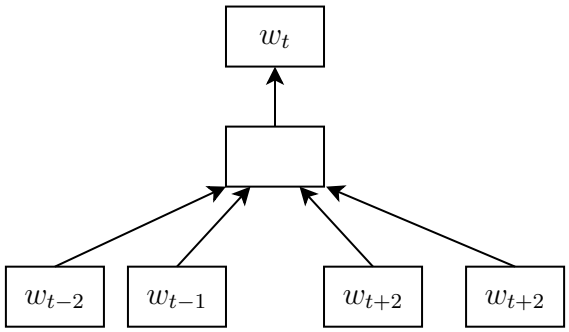
$$J(\boldsymbol{\Theta}) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j}|w_t) \tag{2.7}$$



Obrázek 2.6: *Skip-gram* model.

CBOW (z angl. *continuous bag-of-words*) model předpovídá slovo w_t na základě okolích slov (kontextu), jak lze vidět na obrázku 2.7. Například pro větu *„K snídani jsem měl pomerančový [...] a chléb.“*, doplní na prázdné místo s větší pravděpodobností slovo *„džus“* než *„mléko“*. Model pro trénovací sekvenci velikosti T je optimalizován minimalizací funkce $J(\boldsymbol{\Theta})$, kde m je velikost okénka (viz rovnice 2.8) [47].

$$J(\boldsymbol{\Theta}) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_t|w_{t+j}) \tag{2.8}$$



Obrázek 2.7: *Continuous bag-of-words* model.

Výraz $\log P(o|c)$ je velmi náročný na výpočet, proto se u modelu *Skip-gram* i CBOW používá technika zvaná **negativní samplování**. Je vybráno K slov (často zvolené v rozsahu od 5 do 20) z unigramového rozdělení ($P_n(w) = U(w)^{3/4}/Z$, experimentálně zjištěné v [47]), která mají malou pravděpodobnost výskytu v kontextu centrálního slova w_t. Cílem je upravit parametry tak, aby se pravděpodobnost výskytu pro slovo w_t maximalizovala u skutečných okolních slov a minimalizovala u vybraných K slov. Výpočet $\log P(o|c)$ se nahradí výrazem 2.9 [47], kde σ je sigmoida (viz sekce 3.3).

$$\log P(o|c) = \log(\sigma(\mathbf{u}_o^\mathsf{T}\mathbf{v}_c)) + \sum_{k=1}^{K} \log(\sigma(-\mathbf{u}_k^\mathsf{T}\mathbf{v}_c)) \tag{2.9}$$

FastText *FastText* [10] je metoda založená na stejném principu jako dříve zmíněná metoda *word2vec* (též má dvě varianty CBOW a *Skip-gram* a používá negativní samplování pro urychlení výpočtu) s rozdílem, že kromě vektorů pro slova se učí vektory i pro n-gramy složené ze znaků slov (obvykle v rozpětí od 3 do 6 znaků). Tímto způsobem je metoda schopna na rozdíl od metody *word2vec* vytvořit vektory i pro slova, která nebyla přítomna v trénovacích datech.

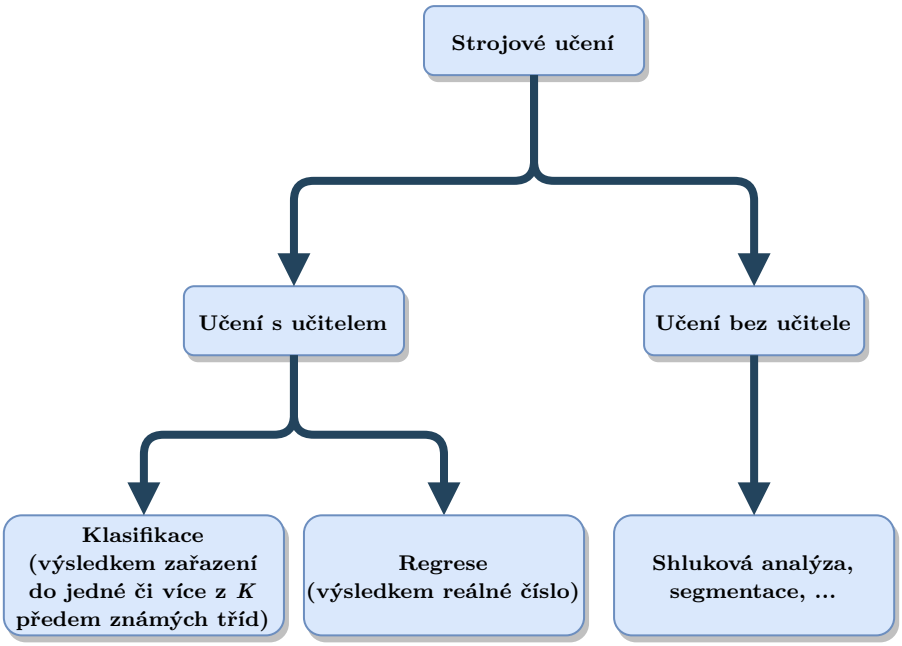
Kontextualizované slovní vektory

Kontextualizované (též dynamické) slovní vektory (angl. *contextual word embeddings*) reprezentují každé slovo na základě všech prvků sekvence a pro stejná slova mohou tvořit různé slovní vektory. Kontextualizované slovní vektory specifikují funkci, jejímž vstupem je sekvence textu a která každému slovu v sekvenci přiřadí vektor [25, 26]. Nejčastěji jsou tyto vektory získávány z modelů založených na architektuře neuronových sítí známých jako *Transformers* [63]. Příkladem je **BERT** [18] (z angl. *Bidirectional Encoder Representations from Transformer*) nebo **ELMo** [52] (z angl. *Embeddings from Language Models*).

3 Metody strojového učení pro klasifikaci

Strojové učení (ML, z angl. *machine learning*) je důležitá podoblast **umělé inteligence** (AI, z angl. *artificial intelligence*), která umožňuje počítačům řešit problémy, k jejichž řešení nebyly explicitně naprogramovány. Pojmem strojové učení se označují procesy, kterými výpočetní systém mění svoje znalosti, strukturu nebo chování na základě analýzy podnětů z prostředí tak, aby v čase rostl jeho výkon [4].

Techniky strojového učení lze rozdělit na dvě hlavní skupiny (viz obrázek 3.1). První skupinou je **učení s učitelem** (angl. *supervised learning*), kde jsou kromě vstupních dat poskytnuta i jejich označení (např. v případě klasifikace, tj. rozdělení do jednotlivých tříd). Druhou je **učení bez učitele** (angl. *unsupervised learning*), kde jsou k dispozici pouze vstupní data [4].



Obrázek 3.1: Rozdělení strojového učení podle typu učení.

Detekce polarity textu se běžně řeší jako klasifikační úloha, proto se jí bude tato kapitola nadále zabývat podrobněji. **Klasifikace** je třídění či zařazování daných objektů (vstupních dat, v NLP obvykle textu) do různých tříd. V analýze sentimentu se klasifikace například používá pro detekci polarity k přiřazení textu do tříd *pozitivní*, *negativní* a případně *neutrální*.

Modely strojového učení, tzv. klasifikátory, se snaží najít vhodnou aproximaci funkce $f(x) : X \to Y$, kde X a Y je množina vstupních a výstupních hodnot, která se nazývá **hypotéza** h. Platí tedy $h \approx f$. Rozlišuje se klasifikace **binární**, kdy je počet tříd roven dvěma, a klasifikace do jedné z K tříd (angl. *multi-class*), kde $K > 2$. ***Multi-label*** klasifikace přiřazuje vzorky do více různých tříd najednou [9]. V této kapitole jsou blíže popsány některé z běžně používaných metod pro klasifikaci textu.

3.1 Naivní bayesovský klasifikátor

Naivní bayesovský klasifikátor (NBC, z angl. *naive Bayes classifier*) je jedním ze základních druhů klasifikátorů. Patří do skupiny pravděpodobnostních klasifikátorů založených na aplikaci **Bayesovy věty** a silných (naivních) předpokladů o podmíněné nezávislosti příznaků klasifikovaných vzorků [4].

Nechť je dáno rozhodovacího pravidlo $f : D \to C$, kde D je množina dokumentů, C množina tříd a každou instanci $d \in D$ lze popsat vektorem příznaků $[f_1, f_2, \ldots, f_n]$. Třída c_{NB} vybraná naivním bayesovským klasifikátorem je potom získána dle vztahu 3.1, kde $P(c)$ značí podmíněnou pravděpodobnost třídy c a $P(f_i|c)$ pravděpodobnost výskytu příznaku f_i v dokumentech patřících do třídy c (odvození viz [30]).

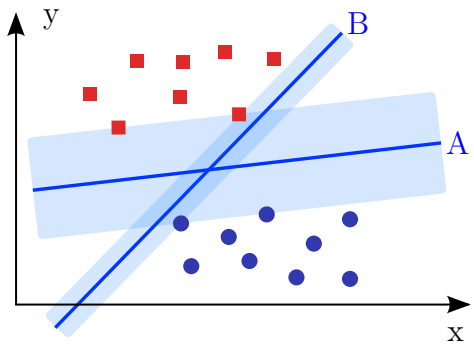
$$c_{\mathrm{NB}} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_i P(f_i|c) \tag{3.1}$$

Naivní bayesovský klasifikátor se velmi často a úspěšně aplikuje na klasifikaci textu (např. třídění článků podle tématu, klasifikace e-mailů na spam/ham, detekce polarity textu atd.). Většinou se používá v kombinaci s modely pro vektorizaci textu BOW a TF-IDF (viz kapitola 2). Jeho hlavní výhodou je jednoduchá implementace a rychlost trénování v porovnání s jinými běžně používanými klasifikátory [30].

3.2 Metoda podpůrných vektorů

Další metodou strojového učení s učitelem sloužící ke klasifikaci je **metoda podpůrných vektorů** (SVM, z angl. *support vector machines*) [15]. Cílem algoritmu je nalézt optimální rozdělující nadrovinu oddělující trénovací data v n-dimenzionálním prostoru příznaků (kde n je počet příznaků) – má co největší vzdálenost od nejbližších vzorků z obou tříd, tzv. **maximální okraj** (angl. *maximum margin*), jak lze vidět na obrázku 3.2. Nadrovina určuje, do

jaké třídy vzorek patří, podle toho, na jaké straně nadroviny leží. Dimenze nadroviny závisí na počtu příznaků. K popisu takové nadroviny stačí pouze nejbližší body zvané **podpůrné vektory** (angl. *support vectors*), které ovlivňují její orientaci a pozici [16].



Obrázek 3.2: Srovnání jedné možné nadroviny B rozdělující data a optimální nadroviny A s maximální vzdáleností od dat v obou třídách (obrázek převzat z [65]).

Nechť T je trénovací množina o m trénovacích vzorcích definována jako $T = \{(\mathbf{x}^{(1)}, y^{(1)}); \ldots; (\mathbf{x}^{(m)}, y^{(m)})\}$, kde $\mathbf{x}^{(i)}$ je vektor příznaků i-tého vzorku ($[x_0^{(i)}, x_1^{(i)}, \ldots, x_n^{(i)}] \in \mathbb{R}^{n+1}, \forall i : x_0^{(i)} = 1$), n je počet příznaků a $y^{(i)}$ třída příslušného vzorku (v případě binární klasifikace platí $y^{(i)} \in \{0, 1\}$). Hypotézu $h_{\boldsymbol{\Theta}}(\mathbf{x})$ lze popsat lineární kombinací vektoru parametrů $\boldsymbol{\Theta} \in \mathbb{R}^{n+1}$ a vstupního vektoru příznaků $\mathbf{x}$ (viz rovnice 3.2). Hypotéza přímo určuje příslušnost vzorků k dané třídě $\hat{y}$ (viz rovnice 3.3). Vektor parametrů $\boldsymbol{\Theta}$ je k nadrovině kolmý. Pro nadrovinu platí $h_{\boldsymbol{\Theta}}(\mathbf{x}) = 0$, pro podpůrné vektory na jedné straně nadroviny $h_{\boldsymbol{\Theta}}(\mathbf{x}) = -1$, na druhé straně potom $h_{\boldsymbol{\Theta}}(\mathbf{x}) = 1$. Při trénování se pro každý trénovací vzorek optimalizují parametry $\boldsymbol{\Theta}$ tak, aby platily nerovnice 3.4 [4].

$$h_{\boldsymbol{\Theta}}(\mathbf{x}) = \boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x} \tag{3.2}$$

$$\hat{y} = \begin{cases} 1, & \text{pokud } \boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x} \geq 0 \\ 0, & \text{jindy} \end{cases} \tag{3.3}$$

$$\boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x}^{(i)} \geq 1, \text{ pokud } y^{(i)} = 1 \tag{3.4a}$$

$$\boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x}^{(i)} \leq -1, \text{ pokud } y^{(i)} = 0 \tag{3.4b}$$

Výše popsaný případ se týká téměř či úplně lineárně separovatelné úlohy (tzv. **lineární SVM**). Pokud to není možné, lze použít techniku zvanou

jádrová transformace (angl. *kernel transformation*). Tato metoda transformuje data do vyšší dimenze, kde je již možné je rozdělit nadrovinou. Transformace je určena **jádrovými funkcemi** (angl. *kernel functions*) [16]. Příkladem jádrové funkce je **gaussovský kernel**, který vyjadřuje podobnost vzorku $\mathbf{x}$ s body z příznakového prostoru, tzv. **landmarky** $\mathbf{l}^{(i)}$ (viz rovnice 3.5) [9].

$$k(\mathbf{x}, \mathbf{l}^{(i)}) = \exp(-\frac{\|\mathbf{x} - \mathbf{l}^{(i)}\|^2}{2\sigma^2}) \tag{3.5}$$

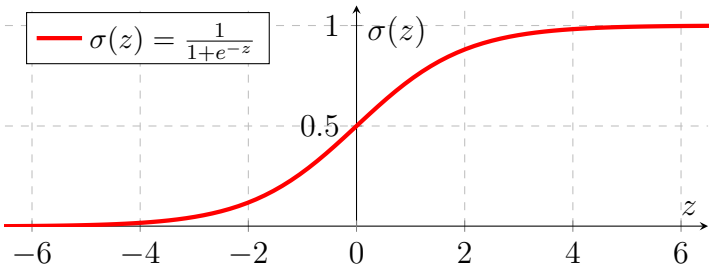
Klasifikaci do více než dvou tříd umožňuje technika ***one-vs-all*** [9]. Natrénuje se K SVM klasifikátorů (kde K je počet tříd), kdy každý poskytuje hypotézu, zda $y = i$, pro $i = 1, 2, \ldots, K$. Trénováním se získá K vektorů parametrů $\boldsymbol{\Theta}^{(1)}, \boldsymbol{\Theta}^{(2)}, \ldots, \boldsymbol{\Theta}^{(K)}$. Následně se vybere třída i s nejvyšší hodnotou $(\boldsymbol{\Theta}^{(i)})^\intercal \mathbf{x}$ (obecně $\max_i h^{(i)}_{\boldsymbol{\Theta}}(\mathbf{x})$).

3.3 Logistická regrese

Logistická regrese je další častou metodou používanou ke klasifikace v oblasti strojového učení s učitelem. Hypotéza logistické regrese $h_{\boldsymbol{\Theta}}(\mathbf{x})$ odhaduje pravděpodobnost, že modelovaná veličina nabude hodnoty 1, pokud je vstupem vektor příznaků $\mathbf{x}$. Proto je nutné, aby byla v rozmezí od 0 do 1 (pravděpodobnost může nabývat pouze hodnot v tomto rozpětí) [30].

K dosažení hodnot hypotézy v požadovaném rozpětí se využívá funkce zvaná **sigmoida** (viz rovnice 3.6, graf viz obrázek 3.3). Sigmoida je speciální druh *logistické funkce*, která mapuje vstupní hodnoty na rozsah od 0 do 1. Funkce je derivovatelná, v okolí 0 má téměř lineární průběh, avšak velmi rychle se hodnoty blíží k 1 pro kladné vstupní argumenty a k 0 pro záporné [30].

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.6}$$



Obrázek 3.3: Graf průběhu funkce sigmoida.

Nechť je trénovací množina definována stejně jako v sekci 3.2 popisující SVM klasifikátor. Logistická regrese se učí z trénovací množiny vektor parametrů $\boldsymbol{\Theta} \in \mathbb{R}^{n+1}$ tak, aby predikovala třídu $\hat{y}$ pro každý vzorek z trénovací množiny s nejmenší chybou od skutečné třídy y. Předpis hypotézy logistické regrese aplikací sigmoidy má tvar 3.7. Vztah 3.8 popisuje rozhodovací pravidlo, podle kterého určí klasifikátor třídu $\hat{y}$ u testovaného vzorku.

$$h_{\boldsymbol{\Theta}}(\mathbf{x}) = \sigma(\boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x}}} \tag{3.7}$$

$$\hat{y} = \begin{cases} 1, & \text{pokud } h_{\boldsymbol{\Theta}}(\mathbf{x}) \geq 0{,}5, \text{ tj. } \boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x} \geq 0 \\ 0, & \text{pokud } h_{\boldsymbol{\Theta}}(\mathbf{x}) < 0{,}5, \text{ tj. } \boldsymbol{\Theta}^{\mathsf{T}}\mathbf{x} < 0 \end{cases} \tag{3.8}$$

K vyjádření správnosti nastavení parametrů $\boldsymbol{\Theta}$ slouží **cenová funkce** (angl. *cost function*) J. Ta má u logistické regrese tvar 3.9 [30], kde m je celkový počet trénovacích vzorků, $y^{(i)}$ skutečná třída trénovacího vzorku i a $h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)})$ hypotézou predikovaná třída z vektoru příznaků i-tého trénovacího vzorku. Tento tvar je též nazýván **křížová entropie** (angl. *cross entropy*).

$$J(\boldsymbol{\Theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)}) + (1 - y^{(i)}) \log(1 - h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)}))\right] \tag{3.9}$$

Pro nastavení optimálních parametrů $\boldsymbol{\Theta}$ je třeba cenovou funkci minimalizovat. K tomu slouží metoda zvaná **gradientní sestup** (angl. *gradient descent*), jehož různé verze jsou popsány např. v [30]. Myšlenkou gradientního sestupu je určit směr, ve kterém funkce klesá, aby bylo dosaženo jejího minima. Směr je dán záporným **gradientem**. Finální rovnici pro aktualizování j-té složky vektoru parametrů $\boldsymbol{\Theta}$ gradientním sestupem (je nutné jednotlivé složky upravovat současně ze stejných hodnot parametrů funkce J) popisuje vztah 3.10 [30], kde $\frac{\partial J(\boldsymbol{\Theta})}{\partial \Theta_j}$ značí parciální derivaci cenové funkce podle Θ_j, $x_j^{(i)}$ je j-tá složka příznakového vektoru i-tého trénovacího vzorku a α **míra učení** (angl. *learning rate*).

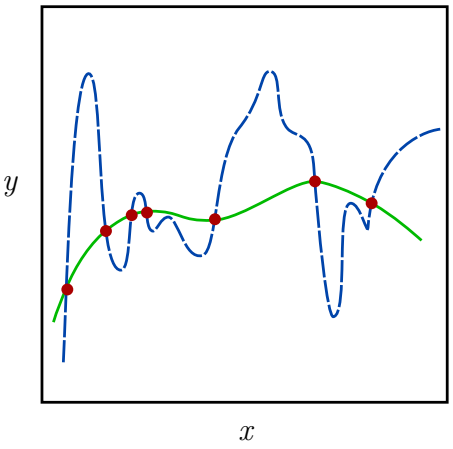
$$\Theta_j = \Theta_j - \alpha\frac{\partial J(\boldsymbol{\Theta})}{\partial \Theta_j} = \Theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)}) - y^{(i)}) \cdot x_j^{(i)} \tag{3.10}$$

Je-li parametr α příliš malý, je metoda gradientního sestupu pomalá, pokud velký, může divergovat (minimum nenalezne, naopak se od něj bude vzdalovat). Parametry jsou upravovány, dokud metoda nedosáhne minima nebo není rozdíl hodnot cenové funkce ve dvou po sobě jdoucích iteracích menší než stanovený práh ϵ. Cenová funkce J je **konvexní**, má tedy pouze jedno minimum, proto metoda gradientního sestupu při správně zvolené míře učení vždy konverguje.

3.3.1 Regularizace

Pokud model perfektně prochází body z trénovací množiny, dojde k **přeučení** (angl. *overfitting*), kdy model špatně predikuje třídy u neznámých testovaných vzorků. Matematicky řečeno model přesně (či velmi blízko) aproximuje trénovací data, bez **generalizace** (predikce v bodech, které nebyly přítomny při trénování modelu), jak ukazuje obrázek 3.4. Aby bylo přeučení předejito, je přidán do rovnice 3.9 tzv. **regularizační člen** [30], čímž se upraví na tvar 3.11, kde n je počet příznaků a λ **regularizační parametr**.

$$J(\boldsymbol{\Theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)}) + (1 - y^{(i)}) \log(1 - h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \Theta_j^2 \tag{3.11}$$



Obrázek 3.4: Vizualizace přeučeného modelu. Červené body reprezentují trénovací data. Modrá přerušovaná křivka představuje příliš přesnou aproximaci (přeučený model), naopak zelená dobrou aproximaci poskytující rozumné předpovědi i pro data, jež nebyla přítomna při trénování modelu (obrázek převzat z [64]).

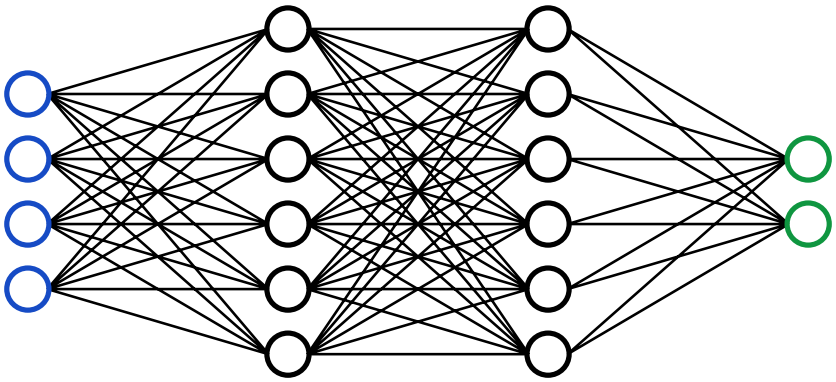
Úprava parametrů $\boldsymbol{\Theta}$ metodou gradientního sestupu bude též modifikována z tvaru 3.10 na 3.12. Regularizační člen penalizuje vysoké hodnoty příznaků, které vedou k přetrénování. Snížením jejich váhy vznikne málo oscilující křivka hypotézy (má hladší průběh), která rozumněji aproximuje data i při použití vysokého stupně polynomu. Člen Θ_0 se nepenalizuje. Je-li regularizační parametr zvolen příliš vysoký, metoda gradientního sestupu nemusí konvergovat a výsledkem může být **nedoučení** (angl. *underfitting*), kdy hypotéza nedokáže spolehlivě predikovat ani trénovací data.

$$\Theta_j = \Theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_{\boldsymbol{\Theta}}(\mathbf{x}^{(i)}) - y^{(i)}) \cdot x_j^{(i)} + \frac{\lambda}{m}\Theta_j\right] \tag{3.12}$$

Pokud je potřeba rozdělit data do více než dvou tříd, lze stejně jako u klasifikátoru SVM popsaném v sekci 3.2 použít metodu ***one-vs-all*** nebo **multinomiální logistickou regresi** vysvětlenou v [30].

3.4 Umělé neuronové sítě

Umělé neuronové sítě (ANN, z angl. *artificial neural networks*) patří mezi nejpoužívanější klasifikační modely současnosti. Základní jednotkou neuronových sítí jsou umělé neurony, které jsou propojeny do komplexní sítě (viz obrázek 3.5). Tento koncept je inspirován lidským mozkem [44].



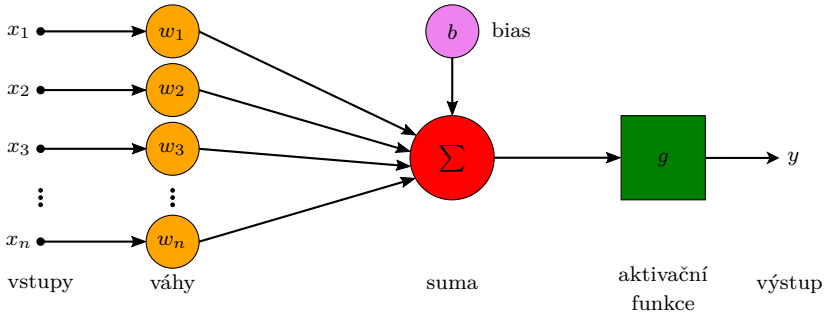
Obrázek 3.5: Příklad umělé neuronové sítě se čtyřmi vstupy, dvěma skrytými vrstvami a dvěma neurony na výstupu.

3.4.1 Dopředná neuronová síť

Dopředná neuronová síť (angl. *feed-forward neural network*) je jedna z nejjednodušších architektur neuronových sítí. Skládá se z jednotlivých vrstev složených z **umělých neuronů**. Strukturu umělého neuronu lze vidět na obrázku 3.6. Neuron kombinuje vstupní vektor $\mathbf{x}$ s vektorem vah $\mathbf{w}$ vypočtením vážené sumy. Předáním součtu reálného čísla b zvaného ***bias*** a vážené sumy jako vstup pro aktivační funkci g je získán výstup (aktivace) y neuronu (viz rovnice 3.13) [30]. Aktivaci celé vrstvy lze spočítat najednou pomocí maticového zápisu, kdy se pro každý neuron i uloží vektor $\mathbf{w}_i$ do matice vah $\mathbf{W}$ a b_i do vektoru $\mathbf{b}$ (viz rovnice 3.14) [30].

$$y = g\left(\sum_{i=0}^{n} x_i w_i + b\right) \tag{3.13}$$

$$y = g(\mathbf{W}^\top \mathbf{x} + \mathbf{b}) \tag{3.14}$$



Obrázek 3.6: Struktura umělého neuronu.

Vrstvě, jejímž vstupem je vektor příznaků, se říká **vstupní vrstva** (angl. *input layer*). Poslední vrstva se nazývá **výstupní vrstva** (angl. *output layer*). Vrstvy, které nejsou vstupní ani výstupní, jsou označovány jako **skryté vrstvy** (angl. *hidden layers*).

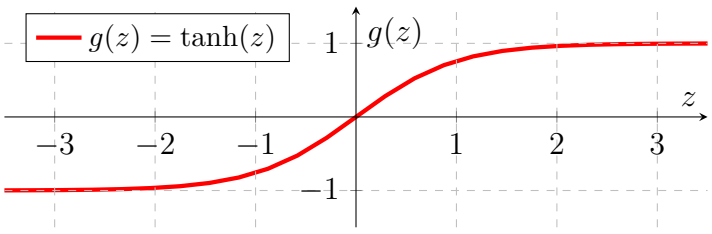
Dopředná síť, ve které je každý neuron propojen se všemi neurony z následující vrstvy, se nazývá **vícevrstvý perceptron**.

3.4.2 Aktivační funkce

Aktivační funkce slouží k získání aktivace neuronů či vrstev. Požaduje se, aby aktivační funkce byla nelineární a spojitě diferencovatelná [31]. Běžně používanou funkcí je **sigmoida**, která byla popsána v sekci 3.3, existují však i jiné aktivační funkce [30].

Hyperbolický tangens (viz rovnice 3.15, graf viz obrázek 3.7) je podobný sigmoidě, ale obor hodnot je od -1 do 1, kdežto u sigmoidy od 0 do 1. Hyperbolický tangens se často používá u skrytých vrstev [31].

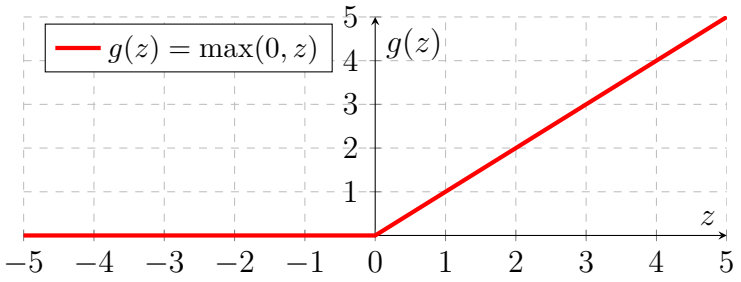
$$g(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{3.15}$$



Obrázek 3.7: Graf průběhu funkce hyperbolický tangens.

Aktivační funkce **ReLU** (z angl. *rectified linear unit*) (viz rovnice 3.16, graf viz obrázek 3.8) se podobně jako hyperbolický tangens nejčastěji používá u skrytých vrstev [31].

$$g(z) = \max(0, z) \tag{3.16}$$



Obrázek 3.8: Graf průběhu funkce ReLU.

Pro výstupní vrstvu se často používá **softmax** funkce (viz rovnice 3.17), která převádí hodnoty vstupního vektoru $\mathbf{z} \in \mathbb{R}^K$ na pravděpodobnostní rozdělení (součet výstupních hodnot softmax funkce je roven jedné, každá hodnota je v rozsahu od 0 do 1) [31].

$$g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.17}$$

3.4.3 Trénování neuronových sítí

Cílem trénování neuronových sítí je získat parametry $\mathbf{W}^{(i)}$ a $\mathbf{b}^{(i)}$ pro každou vrstvu i tak, aby neuronová síť predikovala třídu $\hat{y}$ pro všechny vzorky z trénovací množiny s nejmenší chybou od skutečné třídy y. Proces učení je velmi podobný postupu u logistické regrese popsanému v sekci 3.3.

Nejprve se vypočte aktivace všech neuronů v celé síti **dopředným šířením** (angl. *forward propagation*). K posouzení správnosti nastavení parametrů $\boldsymbol{\Theta}$ neuronové sítě, tvořených *biasy* a vektory vah všech neuronů v síti, slouží **cenová funkce** J. Ta má tvar 3.18 [31] (včetně regularizačního členu sloužícího k zamezení přeučení – viz podsekce 3.3.1), kde m je počet vzorků, λ regularizační parametr, L počet vrstev, K počet výstupních neuronů, s_l počet neuronů v l-té vrstvě, $y_k^{(i)}$ příslušnost i-tého trénovacího vzorku ke k-té třídě (0 znamená nepřísluší, 1 přísluší), $\Theta_{ji}^{(l)}$ váha i-tého vstupu (synapse) j-tého neuronu v l-té vrstvě a $(h_{\boldsymbol{\Theta}}(x))_i$ značí i-tý výstup, platí $h_{\boldsymbol{\Theta}}(x) \in \mathbb{R}^K$. Cenová funkce v tomto tvaru se nazývá **křížová entropie** (angl. *cross entropy*). Pro klasifikaci do více tříd v kombinaci s aktivační funkcí softmax

ve výstupní vrstvě je možné cenovou funkci i dále zjednodušovat [30]. Jiné používané cenové funkce lze najít např. v [31].

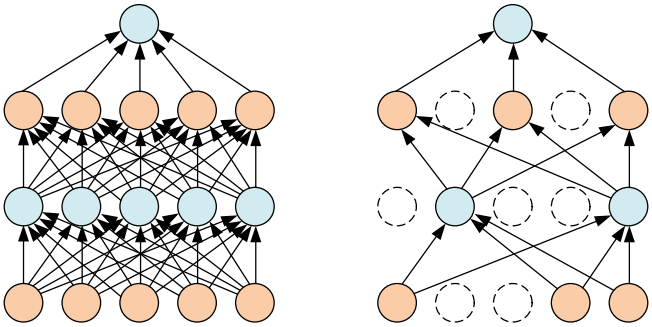
$$J(\mathbf{\Theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_{\mathbf{\Theta}}(\mathbf{x}^{(i)}))_k + (1 - y_k^{(i)}) \log(1 - (h_{\mathbf{\Theta}}(\mathbf{x}^{(i)}))_k)\right] + \\ + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_{ji}^{(l)})^2 \quad (3.18)$$

Dále je cenová funkce minimalizována metodou **gradientního sestupu** (viz sekce 3.3). Jelikož neuronové sítě mají obecně daleko větší počet parametrů oproti logistické regresi, je analytický výpočet gradientu výpočetně náročný. Metoda zvaná **zpětné šíření** (angl. *backpropagation*) umožňuje numerický výpočet derivace $J(\mathbf{\Theta})$ [30]. Metoda je založená na uplatnění pravidla o derivaci složené funkce. Je-li dána funkce $f(x) = u(v(w(x)))$, její derivace podle proměnné x se spočte dle vztahu 3.19 [30].

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \frac{\mathrm{d}u}{\mathrm{d}v} \cdot \frac{\mathrm{d}v}{\mathrm{d}w} \cdot \frac{\mathrm{d}w}{\mathrm{d}x} \quad (3.19)$$

Zpětným šířením je vypočten gradient složený z parciálních derivací cenové funkce, podle kterého se následně upravují hodnoty parametrů, což vede k minimalizaci cenové funkce (blíže popsáno v sekci 3.3) [9, 30].

K potlačení přeučení u neuronových sítí lze též využít techniku zvanou ***dropout*** [59], která při trénování dočasně odstraní neuron společně se všemi spojeními do něj přicházejícími a z něj vycházejícími (viz obrázek 3.9). Ponechání či odstranění neuronu závisí na zadaném parametru p, který udává, s jakou pravděpodobností se má daný neuron odstranit.



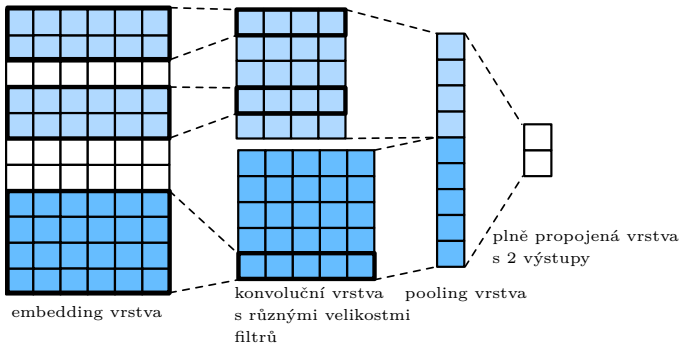
Obrázek 3.9: Vlevo příklad standardní neuronové sítě se dvěma skrytými vrstvami a vpravo redukované sítě po provedení techniky *dropout* (obrázek převzat z [39]).

3.4.4 Další architektury neuronových sítí

V této části jsou zmíněny některé další architektury neuronových sítí.

Konvoluční neuronová síť

Další architekturou neuronových sítí je **konvoluční neuronová síť** (CNN, z angl. *convolutional neural network*) [35]. Ta si sama hledá ve vstupních datech vzory, podle kterých provede klasifikaci, pomocí konvolučních operací. Stejně jako vícevrstvý perceptron je tato architektura neuronové sítě složená z více vrstev, navíc však obsahuje **konvoluční vrstvu**, která provádí konvoluční operace na výstup z předchozí vrstvy (**filtr** definuje velikost oblasti, ze které se konvoluce počítá) a ***pooling* vrstvu**, která snižuje počet příznaků. Obvykle se vícekrát opakuje konvoluční a *pooling* vrstva, následují plně propojené vrstvy (ukázka jednoduché architektury viz obrázek 3.10). Pro úlohy zpracování přirozeného jazyka je potřeba též ***embedding* vrstva**, která obsahuje (či vytváří) slovní vektory pro jednotlivá slova. Konvoluční neuronové sítě se často kromě úloh zpracování přirozeného jazyka používají ke klasifikaci obrázků a videí [31].



Obrázek 3.10: Ukázka schématu architektury konvoluční neuronové sítě.

Rekurentní neuronová síť

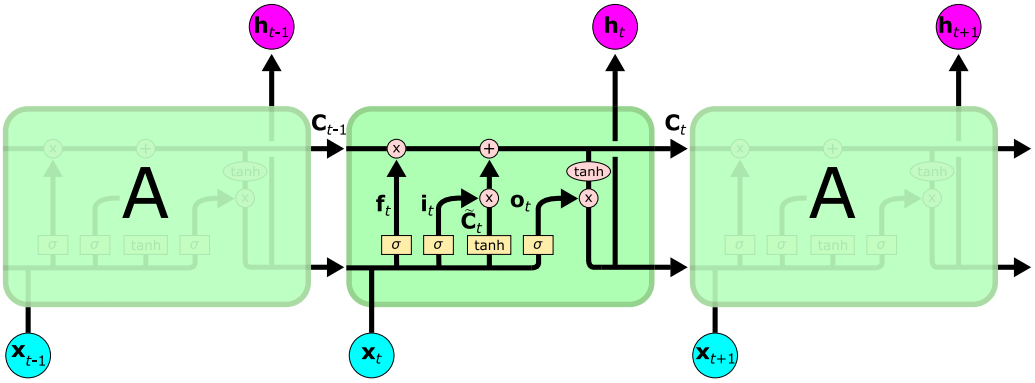
Poslední zmíněnou architekturou, navrženou pro zpracování sekvenčních dat, je **rekurentní neuronová síť** (RNN, z angl. *recurrent neural network*) [19]. Na rozdíl od vícevrstvého perceptronu, který požaduje vstupy fixní délky, mohou rekurentní neuronové sítě zpracovávat sekvence různých délek. V případě této architektury jsou jako vstupy neuronů navíc použity skryté stavy neuronů z předcházející vrstvy. To simuluje paměť a zároveň postihuje fakt, že výstup může záviset na předchozích hodnotách sekvence.

LSTM (z angl. *long short-term memory*) [27] je rozšíření tohoto konceptu, který dokáže lépe postihnout dlouhodobé závislosti v sekvenci. Buňky

LSTM obsahují tři brány (angl. *gates*) – vstupní (angl. *input*), výstupní (angl. *output*) a mazací (angl. *forget*). Tyto brány rozhodují, jaké informace se mají uložit a jaké vymazat (struktura viz obrázek 3.11). LSTM produkuje novou hodnotu vnitřního stavu buňky $\mathbf{C}_t$ a skrytého stavu $\mathbf{h}_t$ ze vstupu $\mathbf{x}_t$, předchozího stavu buňky $\mathbf{C}_{t-1}$ a skrytého stavu $\mathbf{h}_{t-1}$, jak ukazují rovnice 3.20 [49], kde $\odot$ značí násobení po prvcích, $\mathbf{b}$ vektor *biasů*, $\mathbf{W}$ matici vah, $\mathbf{i}_t$ výstup vstupní brány, $\mathbf{f}_t$ výstup mazací brány (hodnoty v rozmezí od 0 – vše smazat – do 1 – vše ponechat), $\mathbf{o}_t$ výstup výstupní brány, $\tilde{\mathbf{C}}_t$ vektor informací, které mohou být přidány, a σ je funkce sigmoida.

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \tag{3.20a}$$
$$\mathbf{i}_t = \sigma(\mathbf{W}_i \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \tag{3.20b}$$
$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{W}_C \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_C) \tag{3.20c}$$
$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t \tag{3.20d}$$
$$\mathbf{o}_t = \sigma(\mathbf{W}_o \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \tag{3.20e}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t) \tag{3.20f}$$

Často se též využívá její obousměrná varianta (BiLSTM, z angl. *bidirectional long short-term memory*) [22], která má stejnou architekturu jako standardní LSTM, obsahuje však další vrstvu, která prochází vstupní sekvenci odzadu dopředu. Tím je navíc modelována závislost daného výstupu i na následujících prvcích sekvence.



Obrázek 3.11: Opakující se buňky LSTM (obrázek převzat a upraven z [49]).

3.5 Metriky pro vyhodnocení klasifikace

V této sekci jsou představeny základní metriky pro vyhodnocení kvality klasifikátorů – úspěšnost, přesnost, úplnost a F-míra.

Úspěšnost (angl. *accuracy*) je definována jako počet správně vyhodnocených vzorků děleno celkovým počtem vzorků (viz rovnice 3.21). Tato metrika je nejjednodušší a nejpoužívanější, není však vhodná pro trénovací množiny, kde je počet vzorků v jednotlivých třídách nevyvážený. Pokud je například v trénovací množině 90 % vzorků označeno jako třída 1, klasifikátoru stačí pouze všechna data zařadit do třídy 1, aby dosáhl 90% úspěšnosti. Proto je vhodné použít i jiné metriky.

$$\text{accuracy} = \frac{\text{správně klasifikované vzorky}}{\text{celkový počet vzorků}} \tag{3.21}$$

Pro definování dalších metrik je nutné vysvětlit pojmy **pravdivě pozitivní** (TP, z angl. *true positive*), **pravdivě negativní** (TN, z angl. *true negative*), **falešně pozitivní** (FP, z angl. *false positive*) a **falešně negativní** (FN, z angl. *false negative*). Matice záměn (viz tabulka 3.1) ukazuje příslušné rozdělení podle různých případů klasifikace.

Skutečná třída / *Predikovaná třída*	**1**	**0**
1	TP	FP
0	FN	TN

Tabulka 3.1: Matice záměn.

Přesnost (angl. *precision*) je poměr počtu správně predikovaných pozitivních vzorků ku celkovému počtu vzorků predikovaných jako pozitivní (viz rovnice 3.22). **Úplnost** (angl. *recall*) je poměr počtu správně klasifikovaných pozitivních vzorků ku celkovému počtu skutečně pozitivních vzorků (viz rovnice 3.23). Nakonec **F-míra** (angl. *F1-score*) je definována jako harmonický průměr přesnosti a úplnosti (viz rovnice 3.24) [41].

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.22}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.23}$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.24}$$

Uvedené rovnice platí pouze pro binární klasifikaci. Existují však celkem tři způsoby generalizace pro klasifikaci do K tříd: **mikro** (angl. *micro*), **makro** (angl. *macro*) a **vážený** (angl. *weighted*) průměr [62]. Nechť $B(tp, fn, fp, tn)$ je daná metrika vypočtená na základě počtu TP, FN, FP a TN. Mikro-průměrovaná metrika je spočtena ze součtů TP, TN, FP a FN přes všechny třídy (viz rovnice 3.25). Platí, že hodnoty všech tří mikro-průměrovaných metrik

(*recall*, *precision* a F-míra) a metriky *accuracy* jsou stejné. Makro-průměrovaná a vážená-průměrovaná metrika se počítají pro každou třídu zvlášť, poté je z těchto hodnot vypočten v prvním případě aritmetický průměr (viz rovnice 3.26), v druhém vážený průměr (viz rovnice 3.27, kde f_k je počet vzorků patřících do třídy k vydělený počtem všech vzorků). Často se uvádějí metriky pro mikro- i makro-průměr. V případě nevyváženého množství vzorků spadajících do jednotlivých tříd lze totiž očekávat, že se klasifikátor lépe naučí predikovat třídy s větším množstvím vzorků, což více ovlivní makro-průměr než mikro-průměr. Příklad matice záměn pro tři třídy ukazuje tabulka 3.2.

$$B_{\text{micro}} = B\left(\sum_{k=1}^{K} tp_k, \sum_{k=1}^{K} fn_k, \sum_{k=1}^{K} fp_k, \sum_{k=1}^{K} tn_k\right) \tag{3.25}$$

$$B_{\text{macro}} = \frac{1}{K}\sum_{k=1}^{K} B(tp_k, fn_k, fp_k, tn_k) \tag{3.26}$$

$$B_{\text{weighted}} = \sum_{k=1}^{K} f_k B(tp_k, fn_k, fp_k, tn_k) \tag{3.27}$$

Skutečná třída / *Predikovaná třída*	**slon**	**tygr**	**nosorožec**
slon	10	1	3
tygr	1	7	0
nosorožec	4	0	6

Tabulka 3.2: Příklad matice záměn pro tři třídy. Je v ní například vidět, že z celkem 15 fotografií slona klasifikátor nesprávně určil třídu dohromady pětkrát – čtyřikrát jako nosorožce, jednou jako tygra.

4 Mezijazyčná detekce polarity a mezijazyčné transformace

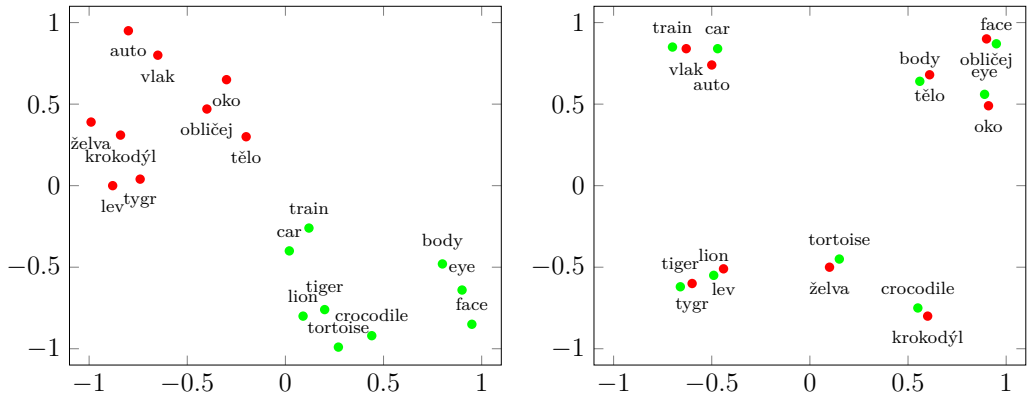
Úkolem **analýzy sentimentu** (SA, z angl. *sentiment analysis*) je detekce, porozumění a extrakce subjektivních informací, např. názorů, emocí a sentimentu [38]. Je to jedno z hlavních a nejvíce se rozvíjejících odvětví zpracování přirozeného jazyka [42].

Detekce polarity je úloha spadající pod analýzu sentimentu, která je běžně využívána společnostmi ke zjištění, zda mají zákazníci pozitivní, neutrální nebo negativní postoj k jejich výrobkům nebo službám [37]. Tato práce se primárně zabývá **mezijazyčnou detekcí polarity**, která využívá mezijazyčných transformací (viz dále), konkrétně k použití anglických dat pro detekci polarity v češtině.

Většina dostupných anotovaných dat určených pro úlohy analýzy sentimentu je v angličtině, dat z jiných jazyků je v porovnání s nimi nedostatek. Jednou z možností k přenosu znalostí mezi jazyky jsou **mezijazyčné transformace**, které dovolují použití anotovaných dat z jazyků s jejich velkým množstvím (angl. *resource-rich languages*) v jazycích, kde je jich nedostatek (angl. *low-resource languages*) [37]. Mezijazyčné transformace také umožňují porovnat význam slov napříč různými jazyky, což je klíčem k strojovému překladu a k mezijazyčnému vyhledávání informací. Příklad je vidět na obrázku 4.1, kde jsou slovní vektory zdrojového i cílového jazyka transformovány do společného prostoru. Dalšími možnostmi přenosu znalostí (resp. využití anotovaných dat v jiném jazyce) jsou strojový překlad [37] a nejnovější mezijazyčné modely založené na architektuře *Transformers* [63].

4.1 Mezijazyčné slovní vektory

Statické slovní vektory jako *word2vec* a *fastText*, popsané v kapitole 2, přinesly velké zvýšení výkonu ve většině úloh zpracování přirozeného jazyka. Dnes jsou též využívány k tvorbě **mezijazyčných slovních vektorů** (CWE, z angl. *cross-lingual word embeddings*). Ty umožňují mezijazyčnou reprezentaci slov, která předpokládá, že vektory představující sémanticky blízká slova v různých jazycích jsou si podobné [56].



(a) Slovní vektory před projekcí

(b) Slovní vektory po projekci

Obrázek 4.1: Příklad vizualizace slovních vektorů pro vybraná česká a anglická slova a jejich projekce do společného prostoru.

Metody mezijazyčných transformací se snaží zarovnat zdrojový prostor na cílový. Používají se tři typy zarovnání dat [56]:

1. **Zarovnání na úrovni slov**: Pro tento způsob používá většina metod dvojici slovo a jeho překlad, který je jednoduše získatelný pro většinu jazyků. Jsou využívány předtrénované slovní vektory, dvojjazyčné slovníky a obvykle lineární transformace.

2. **Zarovnání na úrovni vět**: Paralelní korpus zarovnaný na úrovni vět je dalším typem dat používaných metodami mezijazyčných transformací. Příkladem je Europarl korpus [34].

3. **Zarovnání na úrovni dokumentů**: Paralelní korpus obsahující přeložené dokumenty v různých jazycích (příkladem je Wikipedie, kde je mnoho témat pokryto více jazyky).

Tato kapitola se dále zabývá pouze vybranými metodami zarovnání na úrovni slov.

4.1.1 Metody pro zarovnání na úrovni slov

Rozlišují se tři přístupy pro zarovnání na úrovni slov [56]: přístupy založené na pseudo-vícejazyčných korpusech, dále tzv. *joint* metody a přístupy založené na mapování, kterými se dále tato část kapitoly zabývá.

Přístupy založené na mapování obvykle transformují předtrénované jednojazyčné slovní vektory do společného prostoru pomocí lineárního zobrazení a dvojjazyčných slovníků [46]. Lineární zobrazení umožňuje transfor-

maci mezi dvěma vektorovými prostory prostřednictvím afinních transformací (např. rotace, posun a zrcadlení). Tyto přístupy vychází z pozorování, která ukazují, že rozmístění vektorů slov zdrojového jazyka je po provedení vhodné lineární transformace geometricky velmi podobné rozmístění vektorů slov jejich překladů. Cílem je nalézt transformační matici $\mathbf{W}^{s \to t}$ (dále jen $\mathbf{W}$), která umožní transformovat vektorový prostor zdrojového jazyka s do vektorového prostoru cílového jazyka t [56]. Vynásobením vektoru $\mathbf{x}^s$ slova v původním prostoru transformační maticí $\mathbf{W}$ je získán příslušný vektor $\mathbf{x}^t$ v prostoru cílovém (viz rovnice 4.1). Mezi používané přístupy patří např. regresní a ortogonální metody, které ponechávají vždy jeden prostor nezměněný a druhý na něj mapují a jsou použity v této práci.

$$\mathbf{x}^t = \mathbf{W}\mathbf{x}^s \tag{4.1}$$

Regresní metody

Regresní metody (např. **metoda nejmenších čtverců**) získávají transformační matici $\mathbf{W}$ minimalizací kvadrátu (druhé mocniny) eukleidovské vzdálenosti, tj. střední kvadratické chyby (MSE, z angl. *mean square error*), mezi vektorem $\mathbf{x}_i^s$ zdrojového slova w_i^s transformovaným maticí $\mathbf{W}$ a vektorem $\mathbf{x}_i^t$ jeho překladu w_i^t získaným ze slovníku. Minimalizace je provedena pro n slov zdrojového jazyka $w_1^s, \ldots, w_n^s$ a jejich překladů $w_1^t, \ldots, w_n^t$ (viz rovnice 4.2, která lze přepsat pomocí Frobeniovy normy na tvar 4.3, kde $\mathbf{X}^s$ je matice slovních vektorů slov zdrojového jazyka a $\mathbf{X}^t$ je matice slovních vektorů slov cílového jazyka) [46].

$$\Omega_{\text{MSE}} = \sum_{i=1}^{n} \|\mathbf{W}\mathbf{x}_i^s - \mathbf{x}_i^t\|^2 \tag{4.2}$$

$$\Omega_{\text{MSE}} = \|\mathbf{W}\mathbf{X}^s - \mathbf{X}^t\|_F^2 \tag{4.3}$$

Minimalizaci lze provést buď metodou gradientního sestupu, nebo analyticky dle rovnice 4.4 [56].

$$\mathbf{W} = (\mathbf{X}^{s\intercal}\mathbf{X}^s)^{-1}\mathbf{X}^{s\intercal}\mathbf{X}^t \tag{4.4}$$

Ortogonální metody

Podle [66] by měla být transformační matice **ortogonální**. Matice $\mathbf{W}$ je ortogonální, pokud je čtvercová, její řádky a sloupce tvoří soustavu ortonormálních vektorů, a platí tedy $\mathbf{W}^\intercal\mathbf{W} = \mathbf{W}\mathbf{W}^\intercal = \mathbf{I}$, kde $\mathbf{I}$ je jednotková matice. Optimální transformační matice $\mathbf{W}$ je pak dána vztahem 4.5 [3], kde matice $\mathbf{V}$ a $\mathbf{U}$ jsou získány **singulárním rozkladem** (SVD, z angl. *singular value decomposition*) [21] matice $\mathbf{X}^{t\intercal}\mathbf{X}^s = \mathbf{U}\Sigma\mathbf{V}^\intercal$. Vylepšení ortogonálních

metod spočívá v zachování vlastností z původního prostoru i po transformaci, mimo jiné úhel mezi dvěma slovy zůstává stejný (např. úhel mezi vektory slov *„auto“* a *„motorka“* je totožný s úhlem, který svírají vektory těchto slov po transformaci).

$$\mathbf{W} = \mathbf{V}\mathbf{U}^{\mathsf{T}} \tag{4.5}$$

Další metody pro mezijazyčné transformace

Mezi další metody pro zarovnání na úrovni slov patří **kanonické metody**, které mapují slovní vektory obou jazyků do nového společného prostoru použitím **kanonické korelační analýzy** (CCA, z angl. *canonical correlation analysis*), a ***margin methods***, které místo střední kvadratické chyby optimalizují jinou cenovou funkci [56].

4.2 Další postupy pro přenos znalostí mezi jazyky

Mezi další nástroje pro mezijazyčný přenos znalostí patří **strojový překlad** [37]. Při tomto postupu jsou pro trénování přeložena data ze zdrojového jazyka do cílového. Nevýhodou je potřeba systému pro strojový překlad.

Pokročilejší modely založené na architektuře *Transformers*, jako např. **mBERT** [18], **XLM-RoBERTa** [14] nebo **XLM** [13], jsou přímo trénovány, aby byly mezijazyčné. V porovnání s mezijazyčnými transformacemi vyžadují pro jejich úspěšné trénování a aplikaci velké množství zdrojů (výpočetní výkon a paměť).

4.3 Související práce

V této části jsou zmíněny některé práce, které jsou relevantní k této bakalářské práci a věnují se podobnému tématu, tj. detekci polarity nebo analýze sentimentu, a používají podobné postupy nebo metody. Důkladným prohledáním příslušné literatury nebyly nalezeny žádné související práce pro češtinu.

V publikaci [1] autoři použili předtrénované *word2vec* vektory pro angličtinu a nově vytvořené pro čínštinu, všechny dimenze 300. Nástrojem Překladač Google[1] získali 10 000 překladů dvojic pro slovník nutný k mezijazyčným transformacím. Transformační matice byla získána analytickým

[1] `https://translate.google.com/`

řešením metody nejmenších čtverců (viz rovnice 4.4). Jako datové sady, obě tvořené pěti třídami, použili pro angličtinu 700 000 recenzí podniků ze stránky *Yelp* [69], pro čínštinu dataset představený v [36]. Klasifikátor logistická regrese, který značili autoři jako **ANEW**, trénovali pouze na recenzích cílového jazyka (angličtina) a testovali jen na recenzích zdrojového jazyka (čínština), které transformovali transformační maticí. Navazovali na práci představenou v [12], kde autoři použili klasifikátory logistická regrese značené **BWE** a **DAN** (z angl. *deep averaging network*) a dvojjazyčné slovní vektory, které mapují vektory slov obou jazyků so společného prostoru a jsou drahé na získání. Dosažené výsledky ukazuje tabulka 4.1.

Model	**Accuracy**
BWE	30,58
DAN	29,11
ANEW	28,05

Tabulka 4.1: *Accuracy* v procentech pro jednotlivé modely mezijazyčné klasifikace recenzí představené v [1] a [12].

V [11] autoři trénovali RNN model s obousměrnými vrstvami na anglickém datasetu obsahujícím recenze knih, elektroniky a aplikací z Amazonu [43]. Dále model s využitím natrénovaných vah z generického datasetu specializovali trénováním na recenzích restaurací s dvěma třídami z anglických datových sad *Yelp* a *Kaggle*[2]. Použili předtrénované slovní vektory dimenze 100. Testovací data přeložili do angličtiny pomocí nástroje Překladač Google. Tento přístup nevyžaduje slovní vektory ani trénovací data u jazyků testovacích sad. Testovali na datech čtyř jazyků z datové sady *SemEval-2016 Challenge Task 5* [53]. Tabulka 4.2 ukazuje dosažené výsledky.

Jazyk datasetu	**Accuracy**
španělština	84,21
turečtina	74,36
nizozemština	81,77
ruština	85,61

Tabulka 4.2: *Accuracy* v procentech pro model mezijazyčné klasifikace recenzí restaurací v různých jazycích představený v [11].

V [70] autoři testovali mezijazyčnou analýzu sentimentu na datasetu *NLP&CC 2013*[3], který obsahuje recenze ze tří oblastí (kniha, DVD, hudba).

[2]Ke stažení na `https://www.kaggle.com/c/restaurant-reviews/data`.

[3]`http://tcci.ccf.org.cn/conference/2013/index.html`

Pro každou oblast je přítomno 2 000 pozitivních a 2 000 negativních recenzí v angličtině (cílový jazyk) pro trénování a 4 000 recenzí v čínštině (zdrojový jazyk) pro testování. Obsahuje také data nezařazená do jednotlivých tříd. K překladu dat (celých recenzí), s přiřazenými třídami do čínštiny, nezaražených do tříd a testovacích do angličtiny, využili Překladač Google. Trénovali vlastní model slovních vektorů *word2vec* dimenze 50 pro oba jazyky na datech přiřazených i nepřiřazených do tříd. Jako klasifikátor použili rekurentní neuronovou síť (konkrétně obousměrný LSTM model), rozdělený na části pro anglická i čínská data. Navíc každé větě a každému slovu přiřadili reálné číslo určující, jak velký význam má pro klasifikaci. Testovali celkem tři modely podle použití částí jimi navržené architektury: **EN-Attention**, který překládá čínská data do angličtiny a používá pouze část modelu pro angličtinu, **CN-Attention**, který používá pouze část pro čínštinu, a nakonec **BI-Attention**, který spojuje reprezentaci dokumentů v obou částech. Výsledky ukazuje tabulka 4.3. Jejich modely překonaly dříve představené na stejném datasetu.

Model	**Oblast**			**Průměr**
	kniha	**DVD**	**hudba**	
EN-Attention	79,80	82,70	80,80	81,10
CN-Attention	82,00	84,00	80,90	82,30
BI-Attention	82,10	83,70	81,30	82,40

Tabulka 4.3: *Accuracy* v procentech pro jednotlivé modely mezijazyčné klasifikace recenzí představené v [70].

V [7] autoři vytvořili model k tvorbě slovních vektorů (BLSE, z angl. *bilingual sentiment embeddings*), které reprezentují sémantické informace ve zdrojovém (angličtina) i cílovém (španělština, katalánština a baskičtina) jazyce a informace o sentimentu, které jsou dány anotovanými daty pouze ve zdrojovém jazyce. Pro angličtinu použili předtrénovaný model[4] slovních vektorů, pro zbývající jazyky trénovali *word2vec* model dimenze 300 na korpusu z Wikipedie[5]. Pomocí dvou transformačních matic (jedné pro každý prostor) mapovali původní prostory do společného. Ty optimalizovali minimalizací střední kvadratické chyby mezi vektory párů slov ze slovníku (získán z nástroje Překladač Google) transformovanými příslušnými maticemi. Dále provedli klasifikaci sentimentu pomocí dopředné neuronové sítě s dvěma vrstvami, při jejímž trénování (pouze na datech zdrojového jazyka) byly dále upravovány transformační matice. Testování proběhlo na datech cí-

[4]Ke stažení na `https://code.google.com/archive/p/word2vec/`.

[5]`http://attardi.github.io/wikiextractor/`

lového jazyka. Vektory slov v každém vstupu do neuronové sítě byly nejprve zprůměrovány, výsledný vektor následně transformován příslušnou transformační maticí a predikce získána pomocí funkce softmax. K vyhodnocení modelu použili *OpeNER* [2] anglický a španělský dataset a *MultiBooked* [6] katalánský a baskický dataset. Tyto datové sady obsahují hotelové recenze rozdělené do čtyř tříd (*silně pozitivní*, *pozitivní*, *silně negativní*, *negativní*). Testovali také na dvou třídách, kdy zkombinovali silné a slabé třídy. Dosažené výsledky makro F-míry, které ve většině případů překonali dřívější dosažené v jiných publikacích, ukazuje tabulka 4.4.

2 třídy			**4 třídy**		
ES	**CA**	**EU**	**ES**	**CA**	**EU**
74,60	72,90	69,30	41,20	35,90	30,00

Tabulka 4.4: Makro F-míra v procentech pro model mezijazyčné klasifikace hotelových recenzí pro dvě a čtyři třídy při trénování na angličtině a testování na španělštině (ES), katalánštině (CA) a baskičtině (EU) představený v [7].

4.4 Navržené řešení

Na základě podobných prací, představených v sekci 4.3, byly pro účely této práce vybrány modely neuronových sítí (Bi)LSTM a CNN (popis modelů viz kapitola 5) spolu se slovními vektory *word2vec* a *fastText* a dvěma metodami mezijazyčné transformace – ortogonální a metodou nejmenších čtverců. Tyto modely jsou nejprve vyhodnoceny pouze na češtině, poté i s použitím mezijazyčných transformací. V druhém případě jsou testovány na českých datech a trénovány buď jen na anglických, nebo na anglických i českých.

Modely CNN a LSTM byly upřednostněny před pokročilejšími modely založenými na architektuře *Transformers* především z toho důvodu, že trénování pokročilejších modelů vyžaduje v porovnání s modely CNN a LSTM mnohem více zdrojů (paměť, výpočetní výkon). Dalším důvodem je, že s pokročilejšími modely nelze snadno použít mezijazyčné transformace, které byly uvedeny v zadání této práce.

Pro porovnání klasifikace pouze na českých nebo anglických datech jsou použity též základní modely reprezentace dat BOW a TF-IDF a starší klasifikátory SVM, logistická regrese a naivní bayesovský klasifikátor.

5 Experimenty, data a předzpracování

Tato kapitola popisuje výsledky experimentů provedených v rámci této bakalářské práce a k nim příslušná použitá data, předzpracování a modely. Všechny experimenty byly implementovány v programovacím jazyce *Python*.

5.1 Data a předzpracování

Tato sekce krátce popisuje datové sady, slovní vektory a předzpracování, které byly použity v této bakalářské práci.

5.1.1 Datové sady

Pro testování úspěšnosti klasifikace navrženými metodami byly použity tři datové sady (datasety) – jeden český (ČSFD) a dva anglické (SST a IMDb). Všechny dokumenty ze všech datových sad byly podrobeny různým typům předzpracování, podle zvolené kombinace klasifikátoru a reprezentace textu. Všechny modely byly testovány na dvou (byla vypuštěna třída *neutrální*) a třech (pouze u datových sad, které tento počet obsahují) třídách.

CSFD CZ korpus

Tato datová sada[1], představená v [23], obsahuje 91 381 filmových recenzí z Česko-Slovenské filmové databáze[2] (ČSFD), rozdělených do tří tříd – *pozitivní*, *neutrální* a *negativní*.

Stanford Sentiment Treebank (SST)

SST dataset[3] [58] zahrnuje 215 154 frází v celkem 11 855 větách z americké webové stránky *Rotten Tomatoes*[4], která zveřejňuje filmové recenze. Data jsou rozdělena do celkem pěti kategorií, konkrétně *silně pozitivní*, *pozitivní*, *neutrální*, *negativní* a *silně negativní*. Pro potřeby této práce byla třída *silně pozitivní* sloučena s *pozitivní* a *silně negativní* s *negativní*. Navíc byly použity

[1] Ke stažení na `http://liks.fav.zcu.cz/sentiment/`.
[2] `https://www.csfd.cz/`
[3] Ke stažení na `https://nlp.stanford.edu/sentiment/index.html`.
[4] `https://www.rottentomatoes.com/`

dvě trénovací množiny dat, jedna pouze z vět, do další byly k větám přidány i všechny fráze s počtem tokenů větším než tři.

Large Movie review dataset (IMDb)

IMDb datová sada[5] [40] je složená z 50 000 filmových recenzí z webové stránky *Internet Movie Database*[6] (IMDb). Obsahuje 25 000 recenzí určených k trénování a 25 000 k testování rozdělených do dvou tříd – *pozitivní* a *negativní*.

Rozdělení dat

Všechny datové sady jsou rozděleny do tří disjunktních množin – **trénovací** (angl. *train*), **validační** (angl. *valid* nebo také *dev*) a **testovací** (angl. *test*) data. U datové sady SST je již rozdělení k dispozici, u IMDb bylo náhodně vybráno 2 500 recenzí z trénovací sady jako validační data. Pro testovací data u ČSFD datové sady bylo vybráno 20 % recenzí, ze zbytku poté 10 % pro validační a 90 % pro trénovací data – toto rozdělení bylo poskytnuto vedoucím bakalářské práce. Počty pro jednotlivé datové sady jsou uvedeny v tabulce 5.1.

Dataset	Data	Negativní	Pozitivní	Neutrální	Celkem
ČSFD	*train*	21 441 (32,6 %)	22 117 (33,6 %)	22 235 (33,8 %)	65 793
	dev	2 399 (32,8 %)	2 456 (33,6 %)	2 456 (33,6 %)	7 311
	train + dev	23 840 (32,6 %)	24 573 (33,6 %)	24 691 (33,8 %)	73 104
	test	5 876 (32,1 %)	6 324 (34,6 %)	6 077 (33,2 %)	18 277
	celkem	29 716 (32,5 %)	30 897 (33,8 %)	30 768 (33,7 %)	91 381
SST	*train*	3 310 (38,7 %) / 26 179 (27,9 %)	3 610 (42,3 %) / 30 228 (32,3 %)	1 624 (19,0 %) / 37 302 (39,8 %)	8 544 / 93 709
	dev	428 (38,9 %)	444 (40,3 %)	229 (20,8 %)	1 101
	train + dev	3 738 (38,8 %) / 26 604 (28,1 %)	4 054 (42,0 %) / 30 672 (32,4 %)	1 853 (19,2 %) / 37 531 (39,6 %)	9 645 / 94 810
	test	912 (41,3 %)	909 (41,1 %)	389 (17,6 %)	2 210
	celkem	4 650 (39,2 %) / 27 519 (28,4 %)	4 963 (41,9 %) / 31 581 (32,6 %)	2 242 (18,9 %) / 37 920 (39,1 %)	11 855 / 97 020
IMDb	*train*	11 258 (50,0 %)	11 242 (50,0 %)	-	22 500
	dev	1 242 (49,7 %)	1 258 (50,3 %)	-	2 500
	train + dev	12 500 (50,0 %)	12 500 (50,0 %)	-	25 000
	test	12 500 (50,0 %)	12 500 (50,0 %)	-	25 000
	celkem	25 000 (50,0 %)	25 000 (50,0 %)	-	50 000

Tabulka 5.1: Rozdělení dat podle tříd v jednotlivých datových sadách. V závorce je uveden poměr vzorků patřících do dané třídy ku celkovému počtu vzorků v dané množině v procentech. U SST datové sady jsou za lomítkem uvedeny hodnoty při přidání frází do trénovací množiny.

[5] Ke stažení na `https://ai.stanford.edu/~amaas/data/sentiment/`.

[6] `https://www.imdb.com/`

5.1.2 Slovní vektory

V rámci této bakalářské práce byly použity a natrénovány dva modely slovních vektorů – *word2vec* a *fastText*. Pro češtinu byl každý model trénován na třech zdrojích dat: všechny recenze z ČSFD datasetu, dále data z Wikipedie a nakonec spojení předchozích dvou. Navíc byl testován i předtrénovaný *fastText* model[7]. Pro angličtinu byly slovní vektory natrénovány na recenzích spojených dat z IMDb a SST datasetů. Pro finální experimenty byl vybrán *Skip-gram* model (prvotní experimenty ukázaly, že funguje lépe než CBOW), s dimenzí slovních vektorů 300 a minimální četností slov 5, trénován 15 epoch. Pro tvorbu slovních vektorů byly použity knihovny *gensim* [55] a *fastText* [10]. Velikosti slovníků vytvořených slovních vektorů viz tabulka 5.2.

Data	**Velikost slovníku**
ČSFD	52 324
Wikipedie + ČSFD	602 645
Wikipedie	586 603
IMDb + SST	42 999

Tabulka 5.2: Velikosti slovníků u vytvořených slovních vektorů.

5.1.3 Předzpracování dat

Tato sekce popisuje použité předzpracování textu u modelů popsaných dále v této kapitole. U modelů používajících k reprezentaci textu BOW a TF-IDF byla při tokenizaci odstraněna interpunkce a vybrány byly pouze tokeny složené ze dvou a více alfanumerických znaků. Testovalo se ponechání původní velikosti písmen i *lowercasing*.

Předzpracování textu pro neuronové sítě, klasifikátory SVM a logistická regrese v kombinaci se slovními vektory a data pro trénování slovních vektorů bylo stejné. Všechna velká písmena byla převedena na malá, interpunkce ponechána a oddělena od slov mezerami (statistické údaje viz tabulka 5.3). Výjimku v odstranění interpunkce tvořila čísla, pokud bylo interpunkční znaménko jejich součástí, a dále slova, kde nebyla tečka oddělena z pravé strany mezerou a v části textu před tečkou se nevyskytovala diakritika. Použitý nástroj pro tokenizaci *MorphoDiTa* [60] z knihovny *CorPy*[8] totiž takové případy považuje za odkazy, což je často správná předpověď. Chybně ale rozdělí slova, jestliže uživatel v recenzi zapomněl za tečkou udělat mezeru. Pokud je

[7]Ke stažení na `https://fasttext.cc/docs/en/crawl-vectors.html`.

[8]`https://pypi.org/project/corpy/`

navíc v neodděleném slově za tečkou přítomen znak s diakritikou, oddělí od celého výrazu část od prvního takového výskytu (např. *„spokojen.Velkými“* rozdělí na *„spokojen.Velk“* a *„ými“*). Proto se v předzpracovaném textu mohou vyskytovat např. samostatná písmena *„ý“*. Počet tokenů v ČSFD datasetu po tokenizaci, která obsahují interpunkci (a nejsou samostatná interpunkční znaménka), je 0,11 % z celkového počtu tokenů. Nicméně pokud nejsou počítána čísla a opravdové webové odkazy, tak pouze 0,06 %, což je zanedbatelné množství.

Pro IMDb dataset byl použit stejný nástroj, u SST datové sady postačuje slova rozdělit podle mezery, protože jsou již správně odděleny včetně interpunkce. Výsledky u zmíněného předzpracování byly v průměru o 2 až 5 % lepší u všech modelů, než když při předzpracování textu byly ponechány původní velikosti písmen a interpunkce byla odstraněna (viz tabulka B.1). Zlepšení bylo výraznější u modelu *word2vec* než u modelu *fastText*.

Pakliže byl aplikován *lowercasing* a interpunkce odstraněna, byly dosažené výsledky experimentů u některých vybraných modelů v průměru o 1 % horší než při použití finální verze předzpracování.

Dataset	Data	Celkem	Unikátní (1)	Unikátní (5)
ČSFD	*train*	4 058 443	203 706	41 736
	dev	451 133	55 609	7 704
	train + dev	4 509 576	215 711	44 859
	test	1 143 335	98 319	16 562
	celkem	5 652 911	243 989	52 324
ČSFD (2 třídy)	*train*	2 632 912	158 751	30 752
	dev	293 365	41 874	5 339
	train + dev	2 926 277	168 329	33 099
	test	749 024	75 787	11 792
	celkem	3 675 301	191 021	38 889
SST	*train*	163 566 / 1 003 170	16 579 / 16 579	3 426 / 14 388
	dev	21 274	5 038	482
	train + dev	184 840 / 1 024 444	17 611 / 17 611	3 786 / 14 412
	test	45 405	7 929	950
	celkem	227 245 / 1 066 849	19 536 / 19 536	4 475 / 14 553
SST (2 třídy)	*train*	133 555 / 672 563	14 828 / 16 283	2 886 / 14 338
	dev	17 046	4 339	396
	train + dev	150 601 / 689 609	15 769 / 17 122	3 191 / 11 336
	test	7 576	7 055	786
	celkem	185 624 / 724 632	17 573 / 18 772	3 790 / 11 589
IMDb	*train*	6 457 627	94 393	28 752
	dev	710 955	31 403	7 930
	train + dev	7 168 582	99 299	30 298
	test	7 012 672	97 282	29 722
	celkem	14 181 254	143 618	42 390

Tabulka 5.3: Celkový a unikátní počet tokenů s minimální četností 1 a 5 (dané číslem v závorkách) po předzpracování, kdy jsou nahrazena všechna velká písmena malými a interpunkce je ponechána a oddělena od ostatních slov mezerami, v různých datových sadách. U SST datové sady jsou za lomítkem uvedeny hodnoty při přidání frází do trénovací množiny.

5.2 Použité modely

V této sekci jsou popsány modely použité v této práci (navržené v sekci 4.4). Dále bude model reprezentace dat TF-IDF oddělován od BOW modelu s vyjádřením četnosti či binárním reprezentací. Označení jednotlivých modelů podle zvoleného klasifikátoru a reprezentace dat jsou uvedeny v tabulce 5.4.

Značení	Klasifikátor	Reprezentace dat
LogReg–BOW	logistická regrese	bag-of-words
SVM–BOW	SVM	bag-of-words
NB–BOW	naivní bayesovský klasifikátor	bag-of-words
LogReg–TFIDF	logistická regrese	TF-IDF
SVM–TFIDF	SVM	TF-IDF
NB–TFIDF	naivní bayesovský klasifikátor	TF-IDF
LogReg–W2V	logistická regrese	word2vec
SVM–W2V	SVM	word2vec
LogReg–FT	logistická regrese	fastText
SVM–FT	SVM	fastText
BiLSTM–W2V	BiLSTM	word2vec
BiLSTM–FT	BiLSTM	fastText
CNN–W2V	CNN	word2vec
CNN–FT	CNN	fastText

Tabulka 5.4: Značení jednotlivých modelů podle použitého klasifikátoru a reprezentace dat.

5.2.1 Základní modely

Modely pro reprezentaci textu **BOW** a **TF-IDF** byly použity s klasifikátory **logistická regrese**, **lineární SVM** a **naivní bayesovský klasifikátor**. Testovány byly různé kombinace konfigurací (viz tabulka 5.5): ponechání původní velikosti slov / převedení všech velkých písmen na malá, ponechání/odstranění diakritiky, minimální četnost slov 1/5, použití unigramů (sekvence slov délky 1) / unigramů i bigramů (sekvence slov délky 2) / n-gramů složených z písmen v rozsahu 3 až 6, použití/nepoužití binární reprezentace u BOW a TF-IDF (TF část je 0 nebo 1). Pro všechny uvedené klasifikátory a modely reprezentace textu byla využita existující implementace v knihově *scikit-learn* [51]. Po úvodních experimentech s různými hodnotami (konkrétně 0,0001; 0,001; 0,01; 0,1 a 1) byla míra učení u logistické regrese i SVM nastavena na 0,001. Maximální počet iterací byl nastaven na 500 u logistické regrese a 3 000 u SVM, aby oba klasifikátory dokonvergovaly

ve všech případech konfigurací a na všech datových sadách. Optimalizační problém u logistické regrese byl řešen metodou *saga* [17].

Nastavení	Možnosti
velikost písmen (C)	původní (k) lowercasing (l)
diakritika (D)	ponechána (k) odstraněna (r)
minimální četnost slov (M)	1 (o) 5 (f)
n-gramy (G)	unigramy (u) unigramy + bigramy (b) znaky (c)
binární (B)	ano (t) ne (f)

Tabulka 5.5: Nastavení konfigurace modelů BOW a TF-IDF. Například sekvence *„ClDkMoGuBt“* značí použití techniky *lowercasing*, ponechání diakritiky a zvolení minimální frekvence slov 1, unigramů a binární reprezentace.

Klasifikátory **logistická regrese** a **SVM** byly též použity s modely slovních vektorů ***word2vec*** a ***fastText***. Tyto klasifikátory mohou mít na vstupu pouze vektor fixní velikosti, avšak statické slovní vektory popisují každou sekvenci maticí $n \times d$, kde n je počet slov a d dimenze vektoru. Proto bylo nutné vektor vytvořit pro každý vstup zprůměrováním vektorů slov v něm přítomných, čímž se ztrácí část informace. Tento vektor byl navíc normalizován. U modelu *word2vec* byla přeskočena všechna slova nepřítomná v jeho slovníku a pokud se tak stalo u všech v daném vzorku, byl vrácen nulový vektor. U modelu *fastText* bylo využito jeho schopnosti generovat vektor i pro slova, která nejsou součástí jeho slovníku.

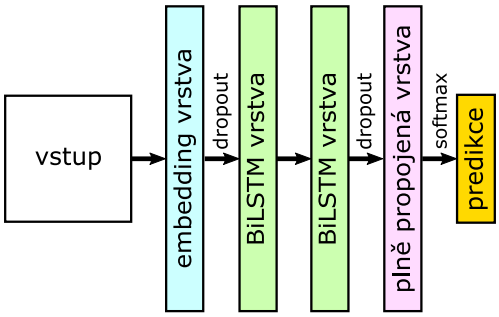
5.2.2 Neuronové sítě

Neuronové sítě byly použity v kombinaci s modely statických slovních vektorů *word2vec* a *fastText*. Do *embedding* vrstvy byly přidány pouze vektory, které přísluší tokenům vybraným pro klasifikaci. U modelu *word2vec* to byly pouze tokeny s minimální četností 5, pokud pro některé neexistovaly slovní vektory, byly náhodně inicializovány hodnotami v rozmezí od -0,25 do 0,25. U modelu *fastText* byly vybrány všechny tokeny, uplatnila se schopnost tohoto modelu vektor pro slova chybějící v jeho slovníku vygenerovat. Slovní vektory v *embedding* vrstvě byly ponechány statické, tzn. již nebylo umožněno je během trénování upravovat, jelikož povolení jejich úpravy při experimentech nepřineslo žádné zlepšení. Byla použita dávka (angl. *batch*) velikosti 32, cenová funkce křížová entropie a softmax jako výstup. Neuronové

sítě byly implementovány v knihovně *PyTorch* [50]. Experimenty popsané dále v této kapitole byly provedeny na následujících modelech neuronových sítí včetně jejich uvedených parametrů (např. míra učení).

LSTM model

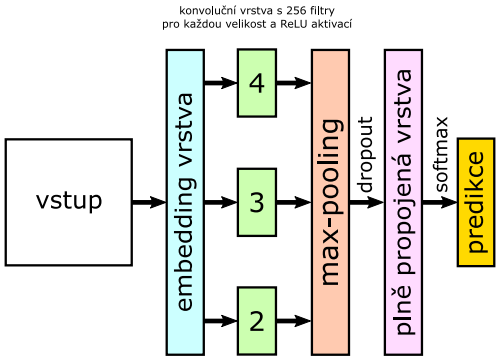
Byl implementován standardní **LSTM** model tvořený *embedding* vrstvou, jednou nebo dvěma LSTM vrstvami s velikostí skrytého stavu **h** 512 a nakonec plně propojenou vrstvou. Technika *dropout* s hodnotou 0,5 byla uplatněna mezi *embedding* a LSTM vrstvou a mezi LSTM a plně propojenou vrstvou. Použila se též její obousměrná varianta (**BiLSTM**). Podobná architektura byla již navržena v [5]. Optimalizace proběhla metodou *Adam* [33] s mírou učení 0,001 (testovány byly i hodnoty 0,0001; 0,01 a 0,1). Vstupy různých velikostí v jedné dávce byly zarovnány na velikost nejdelšího vstupu v dané dávce. Obrázek 5.1 ukazuje architekturu modelu s dvěma obousměrnými LSTM vrstvami.



Obrázek 5.1: Použitý model se dvěma obousměrnými LSTM vrstvami.

CNN model

Dále byla implementována konvoluční neuronová síť – **CNN** model. Architektura, inspirovaná modelem navrženým v [32], se skládá z *embedding* vrstvy, konvoluční vrstvy s filtry velikosti 2, 3 a 4 (256 pro každou velikost), ReLU aktivací a počtem kanálů 300 (dimenze slovních vektorů), dále *max-pooling* a plně propojené vrstvy. Technika *dropout* s hodnotou 0,5 byla uplatněna před plně propojenou vrstvou. Byla použita optimalizační metoda *Adadelta* [68] s mírou učení 0,25 (testovány byly i hodnoty 0,01; 0,1; 0,2; 0,5 a 1) a parametrem *rho* 0,9 (testovány byly i hodnoty 0,5; 0,6; 0,7; 0,8 a 0,95). Věty kratší než největší velikost filtru byly zarovnány na požadovanou délku tzv. *pad* tokenem. Architekturu modelu ukazuje obrázek 5.2.



Obrázek 5.2: Použitý model konvoluční neuronové sítě.

5.3 Vyhodnocení klasifikace

Makro-průměrovaná F-míra (dále jen F-míra) byla zvolena jako hlavní metrika k vyhodnocení klasifikace. Výsledky dalších metrik, konkrétně *accuracy*, *weighted-precision* a *weighted-recall*, lze najít v příloze C ve složce `results`. Nastavení nejlepších konfigurací všech modelů je dáno nejlepším výsledkem F-míry na validačních datech pro modely natrénované na trénovacích datech. Každý model je následně vyhodnocen na testovacích datech a nakonec natrénován na spojených trénovacích a validačních datech a testován na testovacích datech.

U neuronových sítí (konkrétně CNN nebo LSTM) je model s danou konfigurací spuštěn pětkrát a výsledné metriky jsou spočteny jako průměr z jednotlivých běhů. Neuronové sítě jsou vždy trénovány 10 epoch, model z nejlepší epochy podle F-míry je vyhodnocen na testovacích datech. Hodnota nejlepší epochy určuje počet epoch pro následné trénování na spojených trénovacích a validačních datech a vyhodnocení na testovacích datech.

Evaluace modelů trénovaných na anglických datových sadách a ČSFD datasetu redukovaném na dvě třídy proběhla pouze s nejlepšími konfiguracemi získanými na ČSFD datasetu při zachování všech tří tříd. Klasifikace s využitím mezijazyčných transformací byla též provedena jen na nejlepších konfiguracích modelů neuronových sítí získaných na českých datech.

U neuronových sítí je uveden 95% interval spolehlivosti (**konfidenční interval**) [28] pro střední hodnotu μ, který lze získat z rovnice 5.1, kde $\overline{x}$ je průměr, s standardní odchylka, n počet měření (v případě této práce 5), α hladina významnosti (pro 95% interval spolehlivosti je to 0,05), ν stupeň volnosti (platí $\nu = n - 1$) a $t_{1-\frac{\alpha}{2}}(\nu)$ tabulkově daná konstanta.

$$\mu = \overline{x} \pm t_{1-\frac{\alpha}{2}}(\nu) \cdot \frac{s}{\sqrt{n}} \tag{5.1}$$

5.4 Experimenty

Tato sekce popisuje a zhodnocuje dosažené výsledky experimentů, a to jak jednojazyčných, tak i s využitím mezijazyčných transformací.

5.4.1 Výsledky jednojazyčných experimentů

Tato část popisuje získané výsledky pro modely trénované a testované jen na jednotlivých datasetech (resp. jazycích), tj. ČSFD, IMDb, a SST.

Základní modely

Tabulka 5.6 ukazuje výsledky nejlepších konfigurací základních modelů na všech datových sadách.

model	ČSFD		SST		SST (fráze)		IMDb	konfigurace
	3 třídy	2 třídy	3 třídy	2 třídy	3 třídy	2 třídy	2 třídy	
LogReg–BOW	79,80	90,85	53,72	80,39	56,45	82,36	89,29	ClDrMoGbBf
SVM–BOW	78,09	90,81	53,59	79,24	55,28	79,78	89,10	ClDrMoGbBf
NB–BOW	80,58	92,13	55,11	80,34	58,55	83,36	86,80	ClDrMfGbBt
LogReg–TFIDF	80,95	91,65	51,80	79,18	58,59	83,40	90,10	ClDrMfGbBt
SVM–TFIDF	81,33	92,62	52,96	81,92	57,45	83,25	90,25	ClDrMoGbBf
NB–TFIDF	80,91	92,32	48,81	80,67	54,26	83,44	87,67	ClDrMfGbBt
LogReg–TFIDF	81,42	92,08	50,66	81,71	**59,27**	83,74	89,10	ClDrMoGcBf
SVM–TFIDF	**82,26**	**93,29**	**55,32**	82,87	58,09	**83,96**	**90,48**	CkDkMoGcBt
NB–TFIDF	76,27	89,40	47,99	79,95	55,22	83,21	85,63	ClDrMfGcBt
LogReg–W2V	77,71	91,14	53,86	82,97	57,61	82,80	84,98	-
SVM–W2V	77,94	91,58	53,93	83,19	58,10	82,86	87,68	-
LogReg–FT	79,79	93,02	54,57	83,90	58,63	83,36	87,65	-
SVM–FT	79,60	93,11	53,12	**83,91**	58,90	82,86	88,94	-

Tabulka 5.6: Dosažená makro F-míra v procentech u nejlepších konfigurací základních modelů testovaných na různých datových sadách. V případě slovních vektorů byly pro angličtinu použity modely natrénované na datech z IMDb a SST datasetů, pro češtinu na datech z ČSFD datasetu. **Tučně** jsou zvýrazněny nejlepší výsledky v daném datasetu.

V prvních devíti řádcích jsou výsledky klasifikátorů SVM, logistická regrese a naivní bayesovský klasifikátor v kombinaci s modely reprezentace dat BOW a TF-IDF. Ve všech modelech s výjimkou jednoho, který je paradoxně u většiny datových sad nejlepší, bylo nejlepších výsledků dosaženo při převedení velkých písmen na malá a odstranění diakritiky. Vzhledem k tomu, že angličtina neobsahuje diakritiku (či jen v malé míře), lze předpokládat, že výsledky každého modelu na anglických datech by při odstranění i ponechání diakritiky byly stejné či velmi podobné. Nelze vypozorovat, jestli je

lepší minimální četnost slov hodnoty 1 nebo 5. Stejně tak nelze určit, zda je lepší binární reprezentace než vyjádření četnosti u BOW a TF části TF-IDF.

Modely pro reprezentaci dat BOW a TF-IDF fungují lépe, pokud jsou k unigramům přidány i bigramy. Použití n-gramů složených ze znaků místo slov u TF-IDF reprezentace dosahovalo lepších či obdobných výsledků než použití bigramů a unigramů, pouze v kombinaci s naivním bayesovským klasifikátorem byly u většiny datasetů horší (v případě ČSFD datové sady i o více než 4 %). Výsledky dosažené s TF-IDF ve většině případů překonaly klasický BOW, což bude zřejmě dáno tím, že TF-IDF model zahrnuje kromě výskytu také důležitost jednotlivých slov.

Logistická regrese a SVM zpravidla předčily naivní bayesovský klasifikátor. Výjimky lze vidět v kombinaci s BOW, které dokazují, že naivní bayesovský klasifikátor je i přes svoji jednoduchost stále schopen konkurovat pokročilejším modelům. Nelze jednoznačně určit, zda je lepší SVM či logistická regrese, záleží na konkrétním modelu reprezentace dat a datové sadě. Například v kombinaci s klasickým BOW u všech datových sad logistická regrese SVM překonává, naopak v kombinaci s TF-IDF dosahuje SVM většinou lepších výsledků.

Poslední čtyři řádky ukazují výsledky SVM a logistické regrese s modely slovních vektorů *word2vec* a *fastText*. Na českých datech nejlepších výsledků dosáhly oba modely slovních vektorů při trénování na datech z ČSFD, dále na ČSFD a Wikipedii a nejhůře na samostatné Wikipedii, u *fastText* slovních vektorů předtrénovaný jen mírně překonal nejhorší vytvořený model (viz tabulka B.2). Pro anglická data byly použity vytvořené slovní vektory z dat IMDB a SST datových sad. Výsledky jsou podobné a často i horší než u modelů reprezentace textu BOW a TF-IDF. To je pravděpodobně dáno průměrováním jednotlivých vektorů slov v jedné recenzi, což zapříčinilo částečnou ztrátu informace. Klasifikátory s modelem *fastText* zpravidla předčily ty s modelem *word2vec*, zřejmě protože prvně jmenovaný model je schopen generovat vektory i pro neznámá slova (např. u ČSFD datasetu se třemi třídami bylo takto vygenerováno celkem cca 190 000 vektorů, pokud byl model *fastText* trénován pouze na datech z ČSFD). SVM klasifikátor ve většině případů mírně překonal logistickou regresi.

Neuronové sítě

Tabulka 5.7 ukazuje výsledky neuronových sítí se slovními vektory *fastText* a *word2vec*. Stejně jako u základních modelů bylo i u těchto nejlepších výsledků na české datové sadě dosaženo se slovními vektory natrénovanými pouze na datech z ČSFD (viz tabulka B.2). Pro anglická data byly použity

vytvořené slovní vektory z dat IMDB a SST datových sad. Model s dvěma LSTM vrstvami při trénování na trénovacích a testování na validačních datech vždy překonal architekturu s jednou LSTM vrstvou, použití standardních nebo obousměrných LSTM vrstev dosahovalo obdobných výsledků (viz tabulka B.3). Protože průměr výsledků modelu s obousměrnými vrstvami (dále BiLSTM) byl oproti modelu se standardními LSTM vrstvami vyšší a rozptyl nižší, byl pro další výsledky upřednostněn.

model	ČSFD		SST		SST (fráze)		IMDb
	3 třídy	2 třídy	3 třídy	2 třídy	3 třídy	2 třídy	2 třídy
BiLSTM–W2V	82,73 ± 0,25	92,65 ± 0,62	57,30 ± 1,23	84,86 ± 0,43	62,40 ± 0,70	85,80 ± 2,41	90,92 ± 1,47
BiLSTM–FT	**84,92** ± 0,30	**94,29** ± 0,29	**59,75** ± 0,75	**85,26** ± 0,26	**63,77** ± 0,65	**86,92** ± 0,82	**92,57** ± 0,24
CNN–W2V	82,63 ± 0,39	93,41 ± 0,09	53,19 ± 0,59	83,47 ± 0,48	61,13 ± 0,66	**85,92** ± 0,43	91,43 ± 0,31
CNN–FT	83,18 ± 0,14	93,86 ± 0,12	58,53 ± 0,42	**85,05** ± 0,27	59,93 ± 0,61	85,17 ± 0,51	91,45 ± 0,11

Tabulka 5.7: Dosažená makro F-míra v procentech u nejlepších konfigurací modelů neuronových sítí testovaných na různých datových sadách v kombinaci s modely slovních vektorů (natrénované na datech pro angličtinu z IMDb a SST datasetů, pro češtinu z ČSFD datasetu). **Tučně** jsou zvýrazněny nejlepší výsledky v daném datasetu, v případě překrývání intervalu spolehlivosti jsou navíc **podtržené**.

Obecně lze tvrdit, že BiLSTM model překonává CNN, nejspíše z důvodu, že lépe podchycuje dlouhodobější závislosti slov v jednotlivých recenzích. Lepších výsledků je zpravidla dosaženo s modelem slovních vektorů *fastText*, pravděpodobně protože je navíc schopen tvořit vektory pro neznámá slova, jak již bylo zmíněno dříve.

Zhodnocení výsledků jednojazyčných experimentů

Lze vidět, že výsledky dosažené na SST datové sadě jsou hlavně u klasifikace do tří tříd výrazně horší než u ČSFD datasetu. Je to dáno menším počtem vzorků, navíc je jich méně ve třídě *neutrální* oproti ostatním třídám. Proto jsou výsledky v případě dvou tříd, kdy je třída *neutrální* odebrána, výrazně lepší. Přidání frází do trénovací množiny u většiny modelů pomohlo, pravděpodobně i protože se více vyrovnal počet vzorků v jednotlivých třídách (tentokrát bylo ale vzorků ve třídě *neutrální* více než u zbývajících tříd).

Porovnáním nejlepších výsledků u základních modelů a neuronových sítí lze dle očekávání zjistit, že neuronové sítě, které jsou pokročilejší, vždy základní modely překonávají. Rozdíly však nejsou nikterak výrazné, většinou se pohybují okolo 2 %, maximální rozdíl je cca 4 %. Proto nelze jednoznačně tvrdit, že se vždy vyplatí použít neuronové sítě, záleží na konkrétní aplikaci. Implementovat základní modely je totiž snazší a jejich trénování rych-

lejší. Navíc u většiny nejsou potřeba slovní vektory, jejichž získání, pokud již nejsou k dispozici, zabere další čas a prostředky. Obecně model *fastText* překonává *word2vec* a BiLSTM je lepší než CNN.

Porovnání výsledků s jinými pracemi

V této části jsou pro srovnání představeny v tabulce 5.8 některé nejlepší výsledky (SOTA, z angl. *state-of-the-art*) dosažené v jiných publikacích na použitých datových sadách. Jako míra je uvedena *accuracy*. *Accuracy* se u výsledků této práce u většiny modelů liší od makro F-míry zanedbatelně (pro vybrané modely viz tabulka B.4). Výjimkou jsou výsledky získané na třech třídách u SST datasetu (dáno nevyváženým počtem trénovacích vzorků v jednotlivých třídách). Rozdělení dat pro SST a IMDb dataset bylo v uvedených publikacích stejné jako v této práci (při přidání frází se může počet vzorků v trénovací množině u SST datové sady lišit), u ČSFD datové sady rozdílné (82 244 recenzí pro trénování a zbylých 9 137 pro testování).

Modely **XLNet$_{\text{sota}}$**, **XLNet-Large$_{\text{sota}}$** a **Bert_large+ITPT$_{\text{sota}}$**, založené na architektuře neuronových sítí *Transformers*, jsou oproti modelům použitým v této práci modernější, proto je také překonávají. Model **BiLSTM** navržený v této práci je o cca 1,5 % horší než vylepšený model LSTM (**oh-LSTM$_{\text{sota}}$**) pro IMDb dataset, překonává však o více než 2 % **BiLSTM$_{\text{sota}}$** model použitý u SST datasetu na dvou třídách a o více než 4 % **LSTM$_{\text{sota}}$** model u ČSFD datasetu (s jiným rozdělením dat). Taktéž navržený model **CNN** překonává **CNN$_{\text{sota}}$** model u ČSFD datové sady (s rozdílným rozdělením dat) o cca 4,5 %. Pro SST dataset jsou dostupné též výsledky pro pět tříd, pro tři použité v této práci však nebyly žádné nalezeny, stejně tak chybí pro dvě třídy u ČSFD datasetu.

Dataset	Model (SOTA)	Accuracy	BiLSTM	CNN
IMDb	XLNet$_{\text{sota}}$ [67]	96,21	92,57	91,45
	Bert_large+ITPT$_{\text{sota}}$ [61]	95,79		
	oh-LSTM$_{\text{sota}}$ [29]	94,10		
SST (2 třídy)	BiLSTM$_{\text{sota}}$ [5]	82,60	85,26	85,05
SST (fráze) (2 třídy)	XLNet-Large$_{\text{sota}}$ [67]	96,80	86,92	85,92
ČSFD	LSTM$_{\text{sota}}$ [57]	80,50	84,92	83,18
	CNN$_{\text{sota}}$ [57]	78,70		

Tabulka 5.8: *Accuracy* v procentech některých vybraných SOTA modelů testovaných na různých datových sadách, porovnáno s F-mírou v procentech modelů BiLSTM a CNN navržených v této práci.

5.4.2 Mezijazyčné transformace

Mezijazyčné transformace byly provedeny na nejlepších konfiguracích modelů LSTM a CNN pro češtinu v kombinaci se slovními vektory *fastText* natrénovanými na datech z ČSFD datové sady pro češtinu a ze spojených IMDb a SST datových sad pro angličtinu. Nastavení parametrů tedy bylo ponecháno, modely pro klasifikaci se trénovaly znovu od začátku.

Pro získání transformační matice byly použity dvě metody – metoda nejmenších čtverců s analytickým postupem řešení (MSE) a ortogonální metoda (viz kapitola 4). Původně byl počet slov pro tvorbu transformační matice testován v rozmezí od 2 000 do 20 000. Pro konečné experimenty pak byly zvoleny hodnoty 5 000 a 20 000. Slova byla vybrána podle četnosti z natrénovaných slovních vektorů (testovala se i možnost, kdy se zvolilo několik nejčastějších slov a zbytek se vybral náhodně, která však nepřinesla v experimentech zlepšení).

V použitých slovnících překladů získaných z nástroje Překladač Google (dostupné v příloze C, složka `/src/dictionaries`) byla přítomna interpunkce a všechna velká písmena převedena na malá. Provedla se též částečná úprava, mimo jiné v angličtině odstranění určitého členu *the* u 3. stupně přídavných jmen (např. z *„the fastest“* zbylo *„fastest“*). S takto upravenými slovníky bylo dosaženo lepších výsledků než s neupravenými.

Typy mezijazyčné transformace podle trénovací množiny

Při provedených experimentech byly použité modely trénovány buď jen na anglických datech z IMDb či SST datové sady (**čistě mezijazyčná transformace**), nebo na datech anglických i českých (**hybridní mezijazyčná transformace**). Pro každý typ byla navíc provedena transformace z angličtiny do češtiny (**EN→CS**) i naopak (**CS→EN**).

V případě čistě mezijazyčné transformace se trénovalo 10 epoch, po každé epoše byl model vyhodnocen na českých validačních datech. Model z epochy s nejlepší makro F-mírou se následně vyhodnotil na českých testovacích datech. V případě hybridní mezijazyčné transformace se též trénovalo 10 epoch na anglických a českých trénovacích datech, hodnota nejlepší epochy podle F-míry získané vyhodnocením na českých validačních datech následně určila počet epoch pro model trénovaný na datech anglických a českých trénovacích i validačních. Ten byl nakonec vyhodnocen na českých testovacích datech.

Vzhledem k potřebě slovních vektorů pro oba jazyky byly do matice uložené v *embedding* vrstvě přidány k anglickým i české vektory (v obou případech pouze pro tokeny, které se po předzpracování v použitých datech vyskytovaly), čímž se zvětšila paměťová náročnost. Největší matice vektorů

u mezijazyčných transformací (konkrétně hybridní mezijazyčné transformace s IMDb a ČSFD datasetem a dvěma třídami) je ani ne o polovinu větší než největší matice u jednojazyčných experimentů (ČSFD dataset, tři třídy). Matice v dalších případech, zvláště pak u čistě mezijazyčné transformace, kde nejsou potřeba vektory pro tokeny vyskytující se pouze v trénovací množině ČSFD datové sady, jsou ve výsledku velmi podobné nebo i menší než největší matice u jednojazyčných experimentů. Jako druhá možnost se naskýtalo pokaždé měnit matici podle toho, jestli se právě potřebovaly anglické nebo české vektory. Každá taková výměna však zabere určitý čas, navíc nárůst paměťové náročnosti u první možnosti nebyl nikterak výrazný, proto byla upřednostněna první varianta.

5.4.3 Výsledky experimentů s využitím mezijazyčných transformací

V této části jsou uvedeny výsledky experimentů s využitím čistě mezijazyčné transformace i hybridní mezijazyčné transformace, které jsou srovnány s jednojazyčnými výsledky experimentů (viz sloupec **CS** v uvedených tabulkách).

Výsledky čistě mezijazyčné transformace

Tabulka 5.9 ukazuje dosaženou makro F-míru v procentech u modelů CNN a BiLSTM se slovními vektory *fastText* trénovaných na datech z ČSFD datové sady pro češtinu a spojených datech z IMDb a SST datasetů pro angličtinu při užití čistě mezijazyčné transformace (trénováno bylo pouze na anglických datech). Transformační matice byla získána analyticky metodou nejmenších čtverců (MSE) nebo ortogonálně z 5 000 či 20 000 slov a transformace proběhla z angličtiny do češtiny i naopak.

Stejně jako u jednojazyčných výsledků je i zde vidět, že při trénování na SST datasetu a použití tří tříd jsou výsledné hodnoty relativně nízké. To je zřejmě opět zapříčiněno malým počtem vzorků, kde dokonce i při spojení všech dat je trénovací množina menší než testovací z ČSFD datové sady, a také nevyváženým počtem vzorků v jednotlivých třídách. Přidání frází pomohlo méně než u jednojazyčného testování. Nejspíše je to dáno tím, že u SST datasetu je každá recenze složená jen z jedné věty, tudíž i fráze jsou relativně krátké, proto jejich přidání méně ovlivní testování na ČSFD datové sadě, která obsahuje i delší recenze složené z mnoha vět.

Nelze s jistotou tvrdit, že ortogonální metoda je lepší než metoda nejmenších čtverců, neboť např. u SST datové sady, modelu CNN, klasifikaci do tří tříd a transformaci z češtiny do angličtiny je ortogonální metoda o cca 7 %

BiLSTM							
Počet tříd	**CS**	**Dataset**	**Metoda**	**5 000**		**20 000**	
				EN→CS	**CS→EN**	**EN→CS**	**CS→EN**
3	84,92 ± 0,30	SST	MSE	46,06 ± 0,75	**48,91** ± 2,59	45,86 ± 0,92	**47,53** ± 3,72
			ortog.	**47,25** ± 3,10	**48,14** ± 1,63	**49,15** ± 2,10	**49,71** ± 3,03
		SST (fráze)	MSE	46,83 ± 0,60	**46,52** ± 2,89	46,42 ± 0,98	**47,59** ± 4,29
			ortog.	**49,53** ± 2,91	**48,09** ± 1,39	**50,09** ± 2,35	**49,10** ± 2,27
2	94,29 ± 0,29	SST	MSE	82,82 ± 0,71	84,91 ± 0,71	83,37 ± 2,49	82,54 ± 2,49
			ortog.	79,61 ± 2,69	82,41 ± 2,09	81,05 ± 1,88	82,47 ± 2,49
		SST (fráze)	MSE	82,60 ± 2,69	82,57 ± 2,71	83,33 ± 0,93	84,32 ± 0,93
			ortog.	82,42 ± 1,94	83,57 ± 1,25	82,41 ± 2,46	83,14 ± 2,64
		IMDb	MSE	**86,20** ± 1,84	**87,48** ± 1,84	85,17 ± 0,89	**87,98** ± 0,89
			ortog.	**86,89** ± 0,64	**88,25** ± 1,14	**87,59** ± 0,48	**88,57** ± 0,36

CNN							
Počet tříd	**CS**	**Dataset**	**Metoda**	**5 000**		**20 000**	
				EN→CS	**CS→EN**	**EN→CS**	**CS→EN**
3	83,18 ± 0,14	SST	MSE	46,83 ± 0,09	**58,87** ± 1,10	47,12 ± 0,10	**57,60** ± 1,61
			ortog.	**50,32** ± 1,79	50,58 ± 1,10	**51,53** ± 1,42	50,38 ± 1,08
		SST (fráze)	MSE	46,81 ± 0,10	**57,71** ± 1,21	47,18 ± 0,09	**57,63** ± 0,92
			ortog.	**50,07** ± 1,27	51,50 ± 0,86	**51,46** ± 0,63	50,42 ± 0,75
2	93,86 ± 0,12	SST	MSE	86,46 ± 0,19	86,66 ± 0,46	86,99 ± 0,12	86,19 ± 0,65
			ortog.	84,25 ± 1,44	85,46 ± 0,80	86,18 ± 0,30	86,82 ± 0,31
		SST (fráze)	MSE	86,31 ± 0,37	86,47 ± 0,22	86,86 ± 0,17	86,83 ± 0,40
			ortog.	84,80 ± 1,17	86,36 ± 0,40	85,93 ± 0,44	86,41 ± 0,40
		IMDb	MSE	88,05 ± 0,20	86,64 ± 1,35	**88,36** ± 0,10	**88,11** ± 1,72
			ortog.	87,18 ± 0,34	87,81 ± 0,37	**88,16** ± 0,17	**88,99** ± 0,34

Tabulka 5.9: Makro F-míra v procentech pro modely BiLSTM a CNN se slovními vektory *fastText* (natrénované na datech z IMDb a SST datasetů pro angličtinu, z ČSFD datasetu pro češtinu) při použití mezijazyčných transformací s transformační maticí získanou z 5 000 či 20 000 slov metodou nejmenších čtverců nebo ortogonálně a transformací z angličtiny do češtiny (**EN→CS**) i naopak (**CS→EN**). Trénováno jen na datech anglických (sloupec **Dataset**) a vyhodnoceno na českých testovacích. Sloupec **CS** slouží k porovnání s jednojazyčnými výsledky. Pro každý model, počet tříd a směr transformace je nejlepší výsledek zvýrazněný **tučně**, při překrývání intervalu spolehlivosti je navíc **podtržený**.

horší. Naopak pro stejný případ s transformací z angličtiny do češtiny je o cca 4 % lepší. Zvýšení počtu slov pro transformační matici z 5 000 na 20 000 nepřineslo výrazné zlepšení, výsledky jsou velmi podobné, opět v některých případech je o málo lepší první možnost, v jiných druhá. Nakonec ani o směru transformace nelze učinit jednotné rozhodnutí, ale například u metody nejmenších čtverců, modelu CNN, třech tříd a SST datasetu (s frázemi i bez) je výrazně lepší transformace z češtiny do angličtiny nežli naopak. Nedá se tedy říct, že je lepší transformovat prostor s menším počtem dat do prostoru s větším (jak již bylo uvedeno, českých testovacích dat je více než všech dat ze SST datové sady bez frází dohromady použitých pro trénování).

Pro dvě třídy a SST dataset jsou výsledky horší cca o 10 až 15 % u modelu

BiLSTM než u jednojazyčné klasifikace, u CNN pak cca o 7 až 9 %. Za úspěch se dají považovat výsledky získané na IMDb datech, kde je nejlepší výsledek u modelu BiLSTM horší pouze o cca 6 % než jednojazyčný model, u CNN dokonce jen o cca 5 %. Tyto výsledky předčily očekávání a ukázaly, že i model, který nebyl trénován na žádných českých datech, je schopen českou větu klasifikovat poměrně zdárně. Zajímavé je, že na rozdíl od jednojazyčné klasifikace CNN model často překonává BiLSTM model.

Výsledky hybridní mezijazyčné transformace

Výsledky pro hybridní mezijazyčnou transformaci, kdy jsou do trénovací množiny přidána k anglickým datům i česká, ukazuje tabulka 5.10.

BiLSTM							
Počet tříd	**CS**	**Dataset** (+ ČSFD)	**Metoda**	**5 000**		**20 000**	
				EN→CS	**CS→EN**	**EN→CS**	**CS→EN**
3	84,92 ± 0,30	SST	MSE	**84,71** ± 0,47	83,28 ± 0,59	**84,44** ± 0,26	83,17 ± 0,22
			ortog.	**84,91** ± 0,29	**84,71** ± 0,46	**84,55** ± 0,44	**84,38** ± 0,33
		SST (fráze)	MSE	**84,35** ± 0,41	83,26 ± 0,71	**84,39** ± 0,30	83,16 ± 0,19
			ortog.	**84,60** ± 0,36	**84,36** ± 0,67	**84,50** ± 0,25	**84,58** ± 0,35
2	94,29 ± 0,29	SST	MSE	**94,57** ± 0,08	93,84 ± 0,17	**94,39** ± 0,29	93,04 ± 0,48
			ortog.	**94,56** ± 0,15	**94,45** ± 0,18	**94,20** ± 0,30	**94,36** ± 0,21
		SST (fráze)	MSE	94,23 ± 0,23	93,58 ± 0,66	94,19 ± 0,27	93,21 ± 0,29
			ortog.	94,32 ± 0,12	**94,43** ± 0,22	94,14 ± 0,23	**94,47** ± 0,12
		IMDb	MSE	**94,52** ± 0,11	93,80 ± 0,04	**94,54** ± 1,24	93,20 ± 0,37
			ortog.	**94,52** ± 0,21	**94,48** ± 0,24	**94,47** ± 0,20	**94,44** ± 0,07
CNN							
Počet tříd	**CS**	**Dataset** (+ ČSFD)	**Metoda**	**5 000**		**20 000**	
				EN→CS	**CS→EN**	**EN→CS**	**CS→EN**
3	83,18 ± 0,14	SST	MSE	**83,01** ± 0,15	82,37 ± 0,24	82,80 ± 0,17	80,90 ± 1,31
			ortog.	**83,14** ± 0,10	**83,17** ± 0,06	**82,91** ± 0,26	**82,90** ± 0,23
		SST (fráze)	MSE	**83,06** ± 0,11	80,81 ± 0,90	**82,76** ± 0,67	80,57 ± 0,86
			ortog.	**82,97** ± 0,26	82,64 ± 0,38	**82,73** ± 0,58	**82,79** ± 0,34
2	93,86 ± 0,12	SST	MSE	**93,86** ± 0,14	93,35 ± 0,16	93,50 ± 0,20	92,93 ± 0,52
			ortog.	**93,83** ± 0,03	**93,79** ± 0,14	93,51 ± 0,09	93,37 ± 0,11
		SST (fráze)	MSE	93,54 ± 0,26	92,86 ± 0,52	**93,63** ± 0,22	92,99 ± 0,36
			ortog.	93,53 ± 0,25	93,28 ± 0,41	93,55 ± 0,11	93,53 ± 0,15
		IMDb	MSE	**93,76** ± 0,10	93,11 ± 0,14	93,40 ± 0,23	92,40 ± 0,12
			ortog.	**93,81** ± 0,09	**93,77** ± 0,06	93,62 ± 0,17	93,51 ± 0,10

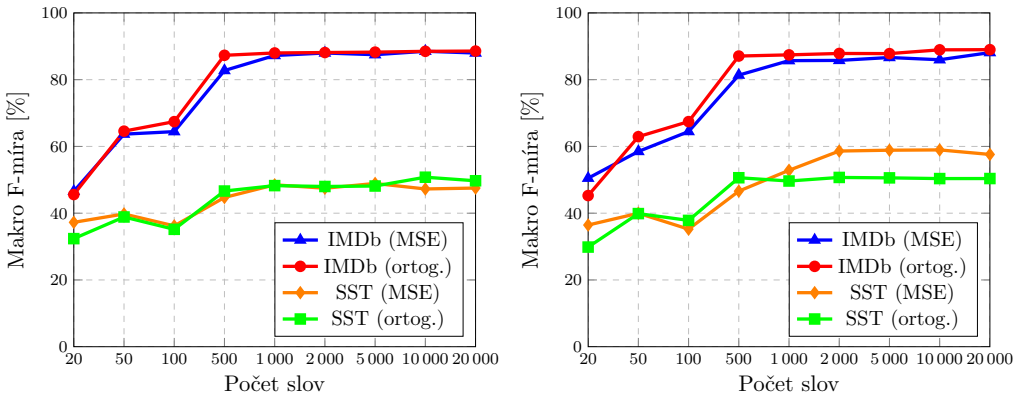
Tabulka 5.10: Makro F-míra v procentech pro modely BiLSTM a CNN se slovními vektory *fastText* (natrénované na datech z IMDb a SST datasetů pro angličtinu, z ČSFD datasetu pro češtinu) při použití mezijazyčných transformací s transformační maticí získanou z 5 000 či 20 000 slov metodou nejmenších čtverců nebo ortogonálně a transformací z angličtiny do češtiny (**EN→CS**) i naopak (**CS→EN**). Trénováno na datech anglických s českými trénovacími a validačními (sloupec **Dataset**) a testováno na českých testovacích. Sloupec **CS** slouží k porovnání s jednojazyčnými výsledky. Pro každý model, počet tříd a směr transformace je nejlepší výsledek zvýrazněný **tučně**, při překrývání intervalu spolehlivosti je navíc **podtržený**.

Výsledky jsou srovnatelné u všech použitých datových sad se získanými z jednojazyčné klasifikace, jak při klasifikaci do dvou, tak i tří tříd. Protože přidání dalších dat z angličtiny nikterak výrazně nepomáhá, lze usuzovat, že trénovací množina ČSFD datasetu je dostatečně velká. U této datové sady také mohlo být téměř dosaženo limitu tzv. *human performance* (zřejmě ani člověk by nebyl schopný přiřadit všechny recenze do správných tříd).

Ani zde nelze jednoznačně určit, zda je lepší ortogonální metoda nebo metoda nejmenších čtverců, použití 5 000 či 20 000 slov pro získání transformační matice nebo transformace z angličtiny do češtiny či naopak. Je však zřejmé, že ortogonální transformace překonává metodu nejmenších čtverců ve všech případech při transformaci z češtiny do angličtiny. Transformace z angličtiny do češtiny v některých případech překonává opačnou, v ostatních je jí velmi podobná. Stejně jako u jednojazyčné klasifikace dosahuje BiLSTM model zpravidla lepších výsledků než CNN model.

Vliv velikosti slovníku na F-míru

Obrázek 5.3 ukazuje závislost makro F-míry na počtu slov pro tvorbu transformační matice při transformaci z češtiny do angličtiny u SST i IMDb datasetu a modelu CNN i BiLSTM při trénování jen na anglických datech.



(a) BiLSTM (b) CNN

Obrázek 5.3: Makro F-míra v závislosti na počtu slov pro tvorbu transformační matice získanou metodou nejmenších čtverců nebo ortogonálně při transformaci z češtiny do angličtiny a trénování na IMDb nebo SST datech u BiLSTM a CNN modelů.

Je vidět, že již výběrem pouhých 500 slov lze dosáhnout srovnatelných výsledků jako s 5 000 slovy, zvláště pak ortogonální transformací. Pro 10 000 a 20 000 slov jsou výsledky velmi podobné 5 000 slov, což již bylo ukázáno

ve výsledcích v tabulkách 5.9 a 5.10. Oproti 500 slov dosahuje 100 slov o cca 20 % horších výsledků u IMDb datové sady, u SST jen o cca 10 %.

Při měření kosinové podobnosti (viz rovnice 2.4) pro počty slov od 500 do 20 000 mezi transformovaným vektorem slova zdrojového jazyka a vektorem jeho skutečného překladu se zjistilo, že při použití menšího počtu roste podobnost u slov vybraných pro tvorbu transformační matice. Naopak při větším počtu roste u slov, která pro tvorbu matice vybrána nebyla. Tyto hodnoty se však příliš nelišily a nejspíše se v konečném důsledku kompenzovaly, proto jsou křivky v grafu od určitého počtu slov téměř konstantní.

Zhodnocení výsledků experimentů s mezijazyčnými transformacemi

Na základě předchozích výsledků lze tvrdit, že přidání anglických dat do trénovací množiny, která již obsahuje česká data, nepomáhá. U čistě mezijazyčné transformace, kde jsou v trénovací množině pouze anglická data a testuje se na češtině, je při použití dostatečného množství anglických dat možné dosáhnout velmi dobrých výsledků. O volbě směru transformace (z češtiny do angličtiny a naopak) i metodě získání transformační matice (ortogonální a metoda nejmenších čtverců) nelze učinit jednoznačné závěry. U vybraných experimentů bylo ukázáno, že i se slovníkem tvořeným 500 slovy pro tvorbu transformační matice je možné (téměř) dosáhnout úrovně nejlepších výsledků a dalším zvětšováním počtu slov hodnoty makro F-míry stagnují či jen velmi mírně rostou.

6 Závěr

V předkládané bakalářské práci, která se jako první zabývá mezijazyčnými transformacemi na českém jazyce pro detekci polarity, jsou v první části popsány různé techniky předzpracování a reprezentace textu pro modely strojového učení, následně základní i pokročilejší metody strojového učení pro klasifikaci a nakonec vybrané metody mezijazyčných transformací.

Dále jsou v práci navrženy a vytvořeny modely pro jednojazyčnou detekci polarity textu, které byly testovány na jedné české (ČSFD) a dvou anglických (SST, IMDb) datových sadách obsahujících recenze filmů rozdělené do dvou nebo tří tříd. Byly použity různé kombinace klasifikátorů a metod reprezentace textu. Povedlo se přiblížit některým *state-of-the-art* výsledkům dosažených podobnými modely, u české datové sady byl dokonce dosavadní nejlepší výsledek překonán (s jiným rozdělením dat). Získané výsledky ukázaly, že neuronové sítě, konkrétně (Bi)LSTM a konvoluční neuronová síť, v kombinaci se slovními vektory *fastText* a *word2vec* překonávají základní modely (SVM, logistická regrese, naivní bayesovský klasifikátor), ale nikterak markantně. Proto je v případě potřeby jednoduchosti či rychlosti možné upřednostnit základní modely. Z výsledků je patrné, že model slovních vektorů *fastText* překonává model *word2vec* a že je výhodné trénovat vlastní slovní vektory, nejlépe na datech z podobné oblasti a se stejným předzpracováním jako u použitých datových sad.

Následně byly nejlepší konfigurace modelů získané na jednojazyčné klasifikaci použity pro experimenty s mezijazyčnými transformacemi. Trénování proběhlo buď čistě na anglických, nebo na anglických i českých datech, testování pak pouze na češtině. Byly testovány dva směry transformace, z češtiny do angličtiny a naopak, a dvě metody vytváření transformační matice, ortogonální a metoda nejmenších čtverců. U trénování pouze na angličtině bylo ukázáno, že použitím dostatečného množství anglických dat je možné dosáhnout velmi dobrých výsledků, které jsou jen o cca 5 až 6 % horší než při trénování na češtině. Tyto získané hodnoty předčily původní očekávání a lze je považovat za úspěch. Trénování na anglických a českých datech nepřineslo oproti trénování jen na češtině žádné výrazné zlepšení.

Všechny předchozí body odpovídají cílům této práce. Další vylepšení by bylo možné důkladnějším předzpracováním dat a úpravou použitých slovníků překladů nebo zlepšením trénovaných modelů, u kterých je možné více experimentovat s parametry. Eventuálně lze také použít modernější modely neuronových sítí a pokročilejší metody mezijazyčných transformací.

Přehled použitých zkratek

AI *artificial intelligence* (umělá inteligence)

ANN *artificial neural network* (umělá neuronová síť)

BiLSTM *bidirectional long short-term memory* (druh rekurentní neuronové sítě s vrstvami procházejícími vstupní řetězec zepředu i zezadu)

BOW *bag-of-words* (model reprezentace textu)

CBOW *continuous bag-of-words* (varianta modelů slovních vektorů *word2vec* a *fastText*)

CCA *canonical correlation analysis* (kanonická korelační analýza)

CNN *convolutional neural network* (konvoluční neuronová síť)

CWE *cross-lingual word embeddings* (mezijazyčné slovní vektory)

ČSFD Česko-Slovenská filmová databáze (použitá pro získání již existující české datové sady s filmovými recenzemi)

FN *false negative* (falešně negativní)

FP *false positive* (falešně pozitivní)

IDF *inverse document frequency* (reprezentuje důležitost slova – část algoritmu TF-IDF)

IMDb *Internet Movie Database* (databáze filmů použitá pro získání již existující datové sady s anglickými filmovými recenzemi)

LSTM *long short-term memory* (druh rekurentní neuronové sítě)

ML *machine learning* (strojové učení)

MSE *mean squarre error* (střední kvadratická chyba, metoda nejmenších čtverců)

NBC *naive Bayes classifier* (naivní bayesovský klasifikátor)

NLP *natural language processing* (zpracování přirozeného jazyka)

RNN *recurrent neural network* (rekurentní neuronová síť)

ReLU *rectified linear unit* (druh aktivační funkce)

SA *sentiment analysis* (analýza sentimentu)

SOTA *state-of-the-art*

SST *Stanford Sentiment Treebank* (datová sada s anglickými filmovými recenzemi)

SVD *singular value decomposition* (singulární rozklad)

SVM *support vector machines* (metoda podpůrných vektorů)

TF *term frequency* (četnost slova – část algoritmu TF-IDF)

TF-IDF *term frequency – inverse document frequency* (algoritmus váhování)

TN *true negative* (pravdivě negativní)

TP *true positive* (pravdivě pozitivní)

Literatura

[1] Abdalla, M. – Hirst, G. Cross-Lingual Sentiment Analysis Without (Good) Translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, s. 506–515, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. Dostupné z: https://www.aclweb.org/anthology/I17-1051.

[2] Agerri, R. et al. OpeNER: Open polarity enhanced named entity recognition. *Procesamiento del Lenguaje Natural*. 2013, 51, s. 215–218.

[3] Artetxe, M. – Labaka, G. – Agirre, E. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, s. 2289–2294, 2016.

[4] Barber, D. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012. ISBN 978-0-521-51814-7.

[5] Barnes, J. – Klinger, R. – Walde, S. Assessing State-of-the-Art Sentiment Models on State-of-the-Art Sentiment Datasets. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, s. 2–12, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-5202. Dostupné z: https://www.aclweb.org/anthology/W17-5202.

[6] Barnes, J. – Badia, T. – Lambert, P. MultiBooked: A Corpus of Basque and Catalan Hotel Reviews Annotated for Aspect-level Sentiment Classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). Dostupné z: https://www.aclweb.org/anthology/L18-1104.

[7] Barnes, J. – Klinger, R. – Walde, S. Bilingual Sentiment Embeddings: Joint Projection of Sentiment Across Languages. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, s. 2483–2493, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1231. Dostupné z: https://www.aclweb.org/anthology/P18-1231.

[8] Bird, S. – Klein, E. – Loper, E. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit.* "O'Reilly Media, Inc.", 2009. ISBN 978-0-596-51649-9.

[9] Bishop, C. M. *Pattern Recognition and Machine Learning.* Springer, 2006. ISBN 978-0387-31073-2.

[10] Bojanowski, P. et al. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics.* 2017, 5, s. 135–146. doi: 10.1162/tacl_a_00051. Dostupné z: https://www.aclweb.org/anthology/Q17-1010.

[11] Can, E. F. – Ezen-Can, A. – Can, F. Multilingual Sentiment Analysis: An RNN-Based Framework for Limited Data. *CoRR.* 2018, abs/1806.04511. Dostupné z: http://arxiv.org/abs/1806.04511.

[12] Chen, X. et al. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics.* 2018, 6, s. 557–570.

[13] CONNEAU, A. – Lample, G. Cross-lingual Language Model Pretraining. In Wallach, H. et al. (Ed.) *Advances in Neural Information Processing Systems*, 32. Curran Associates, Inc., 2019. Dostupné z: https://proceedings.neurips.cc/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf.

[14] Conneau, A. et al. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, s. 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. Dostupné z: https://www.aclweb.org/anthology/2020.acl-main.747.

[15] Cortes, C. – Vapnik, V. Support-vector networks. *Machine learning.* 1995, 20, 3, s. 273–297.

[16] Cristianini, N. – Shawe-Taylor, J. – others. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press, 2000. ISBN 978-0-511-80138-9.

[17] Defazio, A. – Bach, F. – Lacoste-Julien, S. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. In Ghahramani, Z. et al. (Ed.) *Advances in Neural Information Processing Systems*, 27. Curran Associates, Inc., 2014. Dostupné z: https://proceedings.neurips.cc/paper/2014/file/ede7e2b6d13a41ddf9f4bdef84fdc737-Paper.pdf.

[18] Devlin, J. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019.

[19] Elman, J. L. Finding structure in time. *Cognitive science*. 1990, 14, 2, s. 179–211.

[20] Firth, J. R. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*. 1957.

[21] Golub, G. H. – Reinsch, C. *Singular Value Decomposition and Least Squares Solutions*, s. 134–151. Springer Berlin Heidelberg, Berlin, Heidelberg, 1971. doi: 10.1007/978-3-642-86940-2_10. Dostupné z: https://doi.org/10.1007/978-3-642-86940-2_10. ISBN 978-3-642-86940-2.

[22] Graves, A. – Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks*. 2005, 18, 5-6, s. 602–610.

[23] Habernal, I. – Ptáček, T. – Steinberger, J. Sentiment Analysis in Czech Social Media Using Supervised Machine Learning. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, s. 65–74, Atlanta, Georgia, June 2013. Association for Computational Linguistics. Dostupné z: http://www.aclweb.org/anthology/W13-1609.

[24] Harris, Z. S. Distributional structure. *Word*. 1954, 10, 2-3, s. 146–162.

[25] Hewitt, J. *Finding Syntax with Structural Probes* [online]. April 2019. [cit. 2021/04/04]. Dostupné z: https://nlp.stanford.edu//~johnhew//structural-probe.html?utm_source=quora&utm_medium=referral#the-structural-probe.

[26] Hewitt, J. – Manning, C. D. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, s. 4129–4138, 2019.

[27] Hochreiter, S. – Schmidhuber, J. Long short-term memory. *Neural computation*. 1997, 9, 8, s. 1735–1780.

[28] Hogg, R. V. – McKean, J. – Craig, A. T. *Introduction to Mathematical Statistics*. Pearson Education, 2005. ISBN 978-0-13-008507-8.

[29] Johnson, R. – Zhang, T. Supervised and semi-supervised text categorization using LSTM for region embeddings. In *International Conference on Machine Learning*, s. 526–534. PMLR, 2016.

[30] Jurafsky, D. – Martin, J. H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition (2nd Edition)*. Prentice-Hall, Inc., 2009. ISBN 978-0-13-187321-6.

[31] Ketkar, N. – Santana, E. *Deep Learning with Python*. 1. Springer, 2017. ISBN 978-1-4842-2766-4.

[32] Kim, Y. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, s. 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. Dostupné z: `https://www.aclweb.org/anthology/D14-1181`.

[33] Kingma, D. P. – Ba, J. Adam: A Method for Stochastic Optimization. In Bengio, Y. – LeCun, Y. (Ed.) *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. Dostupné z: `http://arxiv.org/abs/1412.6980`.

[34] Koehn, P. A parallel corpus for statistical machine translation. In *Proceedings of MT Summit X*, s. 12–16, 2009.

[35] Le Cun, Y. et al. Handwritten digit recognition: Applications of neural network chips and automatic learning. *IEEE Communications Magazine*. 1989, 27, 11, s. 41–46.

[36] Lin, Y. et al. An Empirical Study on Sentiment Classification of Chinese Review using Word Embedding. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation: Posters*, s. 258–266, Shanghai, China, October 2015. Dostupné z: `https://www.aclweb.org/anthology/Y15-2030`.

[37] Liu, B. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*. 2012, 5, 1, s. 1–167.

[38] Liu, B. – others. Sentiment analysis and subjectivity. *Handbook of natural language processing*. 2010, 2, 2010, s. 627–666.

[39] Liu, G. – Bao, H. – Han, B. A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis. *Mathematical Problems in Engineering*. 2018, 2018.

[40] Maas, A. et al. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, s. 142–150, 2011.

[41] Manning, C. D. – Schütze, H. – Raghavan, P. *Introduction to Information Retrieval*. Cambridge University Press, 2008. ISBN 978-0-521-86571-5.

[42] Mäntylä, M. V. – Graziotin, D. – Kuutila, M. The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. *Computer Science Review*. 2018, 27, s. 16–32.

[43] McAuley, J. et al. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, s. 43–52, 2015.

[44] McCulloch, W. S. – Pitts, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*. 1943, 5, 4, s. 115–133.

[45] Mikolov, T. et al. Efficient Estimation of Word Representations in Vector Space. In Bengio, Y. – LeCun, Y. (Ed.) *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. Dostupné z: `http://arxiv.org/abs/1301.3781`.

[46] Mikolov, T. – Le, Q. V. – Sutskever, I. Exploiting Similarities among Languages for Machine Translation. *CoRR*. 2013, abs/1309.4168. Dostupné z: `http://arxiv.org/abs/1309.4168`.

[47] Mikolov, T. et al. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, s. 3111–3119, 2013.

[48] Mikolov, T. – Yih, W.-t. – Zweig, G. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, s. 746–751, 2013.

[49] Olah, C. Understanding LSTM Networks, 2015. Dostupné z: `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

[50] Paszke, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H. et al. (Ed.) *Advances in Neural Information Processing Systems*, 32. Curran Associates, Inc., 2019. Dostupné z: `https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf`.

[51] Pedregosa, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, 12, s. 2825–2830.

[52] Peters, M. et al. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, s. 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. Dostupné z: `https://www.aclweb.org/anthology/N18-1202`.

[53] Pontiki, M. et al. Semeval-2016 task 5: Aspect based sentiment analysis. In *International workshop on semantic evaluation*, s. 19–30, 2016.

[54] Ramos, J. – others. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, 242, s. 133–142. New Jersey, USA, 2003.

[55] Řehůřek, R. – Sojka, P. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, s. 45–50, Valletta, Malta, May 2010. ELRA.

[56] Ruder, S. – Vulić, I. – Søgaard, A. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*. 2019, 65, s. 569–631.

[57] Sido, J. – Konopík, M. Curriculum Learning in Sentiment Analysis. In Salah, A. A. – Karpov, A. – Potapova, R. (Ed.) *Speech and Computer*, s. 444–450, Cham, 2019. Springer International Publishing. ISBN 978-3-030-26061-3.

[58] Socher, R. et al. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, s. 1631–1642, 2013.

[59] Srivastava, N. et al. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*. 2014, 15, 1, s. 1929–1958.

[60] Straková, J. – Straka, M. – Hajic, J. Open-source tools for morphology, lemmatization, POS tagging and named entity recognition. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, s. 13–18, 2014.

[61] Sun, C. et al. How to fine-tune BERT for text classification? In *China National Conference on Chinese Computational Linguistics*, s. 194–206. Springer, 2019.

[62] Tsoumakas, G. – Katakis, I. – Vlahavas, I. *Mining Multi-label Data*, s. 667–685. Springer US, Boston, MA, 2010. doi: 10.1007/978-0-387-09823-4_34. Dostupné z: https://doi.org/10.1007/978-0-387-09823-4_34. ISBN 978-0-387-09823-4.

[63] Vaswani, A. et al. Attention is All you Need. In Guyon, I. et al. (Ed.) *Advances in Neural Information Processing Systems*, 30. Curran Associates, Inc., 2017. Dostupné z: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[64] *Regularization* [online]. 2020. [cit. 2020/09/16]. Wikipedia Commons. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/0/02/Regularization.svg.

[65] *SVM* [online]. 2020. [cit. 2020/09/16]. Wikipedia Commons. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/f/f2/Svm_intro.svg.

[66] Xing, C. et al. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, s. 1006–1011, 2015.

[67] Yang, Z. et al. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Wallach, H. et al. (Ed.) *Advances in Neural Information Processing Systems*, 32. Curran Associates, Inc., 2019. Dostupné z: https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf.

[68] Zeiler, M. D. ADADELTA: An Adaptive Learning Rate Method. *CoRR*. 2012, abs/1212.5701. Dostupné z: http://arxiv.org/abs/1212.5701.

[69] Zhang, X. – Zhao, J. – LeCun, Y. Character-Level Convolutional Networks for Text Classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, s. 649–657, Cambridge, MA, USA, 2015. MIT Press.

[70] Zhou, X. – Wan, X. – Xiao, J. Attention-based LSTM Network for Cross-Lingual Sentiment Classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, s. 247–256, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1024. Dostupné z: https://www.aclweb.org/anthology/D16-1024.

A Uživatelská příručka

Uživatelská příručka obsahuje základní informace o spuštění programu, který se na DVD nachází ve složce `/src`. K implementaci byl použit programovací jazyk *Python* (konkrétně verze 3.7.3). Program se spustí zadáním příkazu `python main.py [parametry]`. Výsledky běhu programu s popisem konfigurace jsou uloženy v souboru `results.txt`, který je automaticky vytvořen. Při dalším spuštění programu jsou údaje do souboru připsány na jeho konec.

A.1 Povinné parametry

Program požaduje dva povinné parametry, které určují, jaký typ klasifikátoru a reprezentace dat mají být použity (v tomto pořadí). Lze zvolit z 5 možných klasifikátorů:

- `cnn` – Konvoluční neuronová síť.
- `log_reg` – Logistická regrese.
- `lstm` – LSTM.
- `naive_bayes` – Naivní bayesovský klasifikátor.
- `svm` – SVM.

Je možné vybrat ze 4 různých modelů reprezentace dat:

- `bag_of_word` – Bag-of-words model.
- `fasttext` – *FastText* model.
- `tfidf` – TF-IDF model.
- `word2vec` – *Word2vec* model.

A.2 Volitelné parametry

Dále je možné upravovat konfiguraci modelu nepovinnými parametry, které lze kombinovat a zadávat v krátké (např. `-b`) i dlouhé (např. `--binary`) podobě:

- `-b`, `--binary` – Použití binární reprezentace modelu BOW a TF části modelu TF-IDF.
- `-bd`, `--bidirectional` – Použití obousměrné varianty LSTM modelu (BiLSTM).
- `-bg`, `--bigrams` – Přidá k unigramům i bigramy při kombinaci s modely reprezentace dat BOW a TF-IDF. Pokud je zároveň zvolen parametr `-ch`, je tento ignorován.
- `-ch`, `--character` – Použití n-gramů složených z písmen v rozsahu 3 až 6 při kombinaci s modely reprezentace dat BOW a TF-IDF.
- `-d`, `--dataset` – Dataset, na kterém má proběhnout klasifikace:
  - `csfd` – ČSFD datová sada. Výchozí hodnota.
  - `imdb` – IMDb datová sada.
  - `sst` – SST datová sada.
  - `sstp` – SST datová sada s přidáním frází do trénovací množiny.
- `-fd`, `--f_dataset` – Určuje, jaký *fastText* model (jen pokud je tento model zvolen) má být použit podle dat, na kterých byl natrénován:
  - `csfd` – ČSFD datová sada.
  - `en` – IMDb a SST datová sada.
  - `pretrained` – Předtrénovaný český model. Výchozí hodnota.
  - `wiki` – Wikipedie.
  - `wiki_csfd` – ČSFD datová sada + Wikipedie.
- `-h`, `--help` – Vytiskne nápovědu a ukončí program.
- `-lc`, `--lowercase` – Převede všechna velká písmena na malá při kombinaci s modely reprezentace dat BOW a TF-IDF.
- `-min`, `--mid_df` – Nastaví minimální četnost slov na 5 při kombinaci s modely reprezentace dat BOW a TF-IDF.
- `-nl`, `--num_layers` – Udává počet LSTM vrstev. Lze zvolit 1 nebo 2. Výchozí hodnota je 1. Pokud není klasifikátorem LSTM, je parametr ignorován.

- `-nw`, `--num_words` – Počet slov pro tvorbu transformační matice. Výchozí hodnota je 5 000. Pokud není zvolena mezijazyčná transformace, je parametr ignorován.
- `-or`, `--orthogonal` – Použití ortogonální transformace. Pokud není zvolena mezijazyčná transformace, je parametr ignorován.
- `-rd`, `--remove_diacritic` – Odstranění diakritiky při kombinaci s modely reprezentace dat BOW a TF-IDF.
- `-tc`, `--two_classes` – Použití pouze dvou tříd (*pozitivní* a *negativní*) u dané datové sady.
- `-tr`, `--transformation` – Použití mezijazyčné transformace. Pokud není klasifikátorem neuronová síť, je parametr ignorován.
- `-tt`, `--transformation_type` – Typ použité transformace (jen pokud je mezijazyčná transformace zvolena) podle směru a trénovací množiny:
  - `cs_to_en_both` – Transformace z češtiny do angličtiny, trénování na anglických i českých datech.
  - `cs_to_en_only` – Transformace z češtiny do angličtiny, trénování jen na anglických datech. Výchozí hodnota.
  - `en_to_cs_both` – Transformace z angličtiny do češtiny, trénování na anglických i českých datech.
  - `en_to_cs_only` – Transformace z angličtiny do češtiny, trénování jen na anglických datech.
- `-w2vd`, `--w2v_dataset` – Určuje, jaký *word2vec* model (jen pokud je tento model zvolen) má být použit podle dat, na kterých byl natrénován:
  - `csfd` – ČSFD datová sada. Výchozí hodnota.
  - `en` – IMDb a SST datová sada.
  - `wiki` – Wikipedie.
  - `wiki_csfd` – ČSFD datová sada + Wikipedie.

A.3 Omezení

Modely pro reprezentaci textu BOW a TF-IDF nelze zvolit v kombinaci s klasifikátory CNN a LSTM. Modely *word2vec* a *fastText* není možné použít s naivním bayesovským klasifikátorem.

Data, na kterých byly trénovány modely *word2vec* a *fastText*, je nutné zvolit podle jazyku datové sady, na které je prováděna klasifikace. U ČSFD datové sady lze tedy zvolit možnosti `csfd`, `wiki_csfd` nebo `wiki` (v případě modelu *fastText* i `pretrained`), u SST a IMDb datasetů pouze možnost `en`.

Při použití mezijazyčné transformace je nutné jako parametr `--dataset` zvolit anglickou datovou sadu, tedy možnosti `imdb`, `sst` a `sstp`. Při zvolení IMDb datové sady jsou automaticky použity pouze dvě třídy z ČSFD datasetu. Jako parametr představující data, na kterých byly trénovány slovní vektory, je nutné vybrat jednu z možností reprezentující česká data, tedy `csfd`, `wiki_csfd` a `wiki`, v případě modelu *fastText* i `pretrained`.

B Další vybrané výsledky experimentů

Tato příloha ukazuje některé další výsledky experimentů. Na rozdíl od výsledků v textu jsou výsledky u prvních tří tabulek uvedeny při trénování na trénovacích datech a vyhodnocení na validačních datech, podle kterých byly zvoleny konfigurace modelů pro další testování. Tabulka B.1 ukazuje porovnání dvou různých typů předzpracování pro neuronové sítě a data sloužící k trénování modelů slovních vektorů, tabulka B.2 porovnání modelů slovních vektorů podle dat použitých pro jejich trénování a tabulka B.3 porovnání LSTM modelů s různým počtem a typem LSTM vrstev. Tabulka B.4 ukazuje srovnání *accuracy* a makro F-míry u vybraných modelů.

Klasifikátor	**Model slovních vektorů**	**Data použitá pro trénování slovních vektorů**	**A**	**B**
BiLSTM	word2vec	ČSFD	**82,69** ± 0,32	77,44 ± 0,34
		Wikipedie + ČSFD	**81,95** ± 0,14	77,34 ± 0,19
		Wikipedie	**79,36** ± 0,52	71,29 ± 4,68
	fastText	předtrénovaný model	**81,35** ± 0,62	79,56 ± 0,55
		ČSFD	**83,75** ± 0,21	81,39 ± 0,25
		Wikipedie + ČSFD	**83,68** ± 0,21	80,95 ± 0,25
		Wikipedie	**82,38** ± 0,32	79,68 ± 0,71
CNN	word2vec	ČSFD	**81,95** ± 0,22	76,38 ± 0,27
		Wikipedie + ČSFD	**81,31** ± 0,14	77,01 ± 0,16
		Wikipedie	**78,22** ± 0,12	74,59 ± 0,15
	fastText	předtrénovaný model	**79,39** ± 0,22	77,73 ± 0,19
		ČSFD	**82,46** ± 0,16	80,03 ± 0,38
		Wikipedie + ČSFD	**82,05** ± 0,24	78,83 ± 0,22
		Wikipedie	**81,02** ± 0,30	77,92 ± 0,31

Tabulka B.1: Dosažená makro F-míra v procentech u modelů neuronových sítí v kombinaci s modely slovních vektorů *word2vec* a *fastText* v závislosti na použitém předzpracování textu pro neuronové sítě a data pro trénování slovních vektorů. **A** značí předzpracování, kdy jsou velká písmena převedena na malá a interpunkce oddělena mezerami. **B** značí předzpracování, při kterém je ponechána původní velikost písmen a odstraněna interpunkce. Testováno na ČSFD datové sadě při použití tří tříd, trénování na trénovacích a testování na validačních datech. **Tučně** jsou zvýrazněny lepší výsledky předzpracování pro daný klasifikátor, model slovních vektorů a data použitá pro trénování slovních vektorů, na základě kterých byla následně vybrána finální verze předzpracování.

Klasifikátor	**Model slovních vektorů**	**Data použitá pro trénování slovních vektorů**	**F-míra**
logistická regrese	word2vec	ČSFD	**77,22**
		Wikipedie + ČSFD	72,44
		Wikipedie	66,78
	fastText	předtrénovaný model	70,82
		ČSFD	**79,36**
		Wikipedie + ČSFD	75,79
		Wikipedie	69,00
SVM	word2vec	ČSFD	**77,36**
		Wikipedie + ČSFD	72,65
		Wikipedie	67,04
	fastText	předtrénovaný model	71,13
		ČSFD	**79,08**
		Wikipedie + ČSFD	75,68
		Wikipedie	68,67
BiLSTM	word2vec	ČSFD	**82,69** ± 0,32
		Wikipedie + ČSFD	81,95 ± 0,14
		Wikipedie	79,36 ± 0,52
	fastText	předtrénovaný model	81,35 ± 0,62
		ČSFD	**83,75** ± 0,21
		Wikipedie + ČSFD	**83,68** ± 0,21
		Wikipedie	82,38 ± 0,32
CNN	word2vec	ČSFD	**81,95** ± 0,22
		Wikipedie + ČSFD	81,31 ± 0,14
		Wikipedie	78,22 ± 0,12
	fastText	předtrénovaný model	79,39 ± 0,22
		ČSFD	**82,46** ± 0,16
		Wikipedie + ČSFD	82,05 ± 0,24
		Wikipedie	81,02 ± 0,30

Tabulka B.2: Dosažená makro F-míra v procentech u modelů různých klasifikátorů v kombinaci s modely slovních vektorů *word2vec* a *fastText* v závislosti na datech použitých pro trénování slovních vektorů. Testováno na ČSFD datové sadě při použití tří tříd, trénování na trénovacích a testování na validačních datech. **Tučně** jsou zvýrazněny nejlepší výsledky pro daný klasifikátor a model slovních vektorů (v případě překrývání intervalu spolehlivosti jsou navíc **podtržené**), na základě kterých byla následně vybrána data pro trénování slovních vektorů pro další experimenty.

Model slovních vektorů	Počet LSTM vrstev	Typ LSTM vrstvy	F-míra
word2vec	1	standardní	81,10 ± 0,48
	2	standardní	**81,26** ± 1,71
	1	obousměrná	81,28 ± 0,22
	2	obousměrná	**82,69** ± 0,32
fastText	1	standardní	83,23 ± 0,19
	2	standardní	**83,28** ± 0,48
	1	obousměrná	83,18 ± 0,19
	2	obousměrná	**83,75** ± 0,21

Tabulka B.3: Dosažená makro F-míra v procentech u LSTM modelu s použitím jedné či dvou standardních nebo obousměrných LSTM vrstev v kombinaci s modely slovních vektorů *word2vec* a *fastText* trénovaných na datech z ČSFD datové sady. Testováno na ČSFD datové sadě při použití tří tříd, trénování na trénovacích a testování na validačních datech. **Tučně** jsou zvýrazněny nejlepší výsledky pro daný model slovních vektorů (v případě překrývání intervalu spolehlivosti jsou navíc **podtržené**), na základě kterých byl následně vybrán počet a typ LSTM vrstev pro další experimenty.

Dataset	Model	F-míra	Accuracy
ČSFD (3 třídy)	BiLSTM-W2V	82,73 ± 0,25	82,95 ± 0,27
	BiLSTM-FT	84,92 ± 0,30	85,02 ± 0,31
	CNN-W2V	82,63 ± 0,39	82,70 ± 0,41
	CNN-FT	83,18 ± 0,14	83,30 ± 0,12
SST (2 třídy)	BiLSTM-W2V	84,86 ± 0,43	84,87 ± 0,43
	BiLSTM-FT	85,26 ± 0,26	85,27 ± 0,26
	CNN-W2V	83,47 ± 0,48	83,48 ± 0,47
	CNN-FT	85,05 ± 0,27	85,06 ± 0,27
SST (fráze) (2 třídy)	BiLSTM-W2V	85,80 ± 2,41	85,88 ± 2,30
	BiLSTM-FT	86,92 ± 0,82	86,93 ± 0,81
	CNN-W2V	85,92 ± 0,43	85,93 ± 0,43
	CNN-FT	85,17 ± 0,51	85,18 ± 0,51
IMDb	BiLSTM-W2V	90,92 ± 1,47	90,92 ± 1,48
	BiLSTM-FT	92,57 ± 0,24	92,57 ± 0,24
	CNN-W2V	91,43 ± 0,31	91,44 ± 0,31
	CNN-FT	91,45 ± 0,11	91,45 ± 0,11

Tabulka B.4: Porovnání *accuracy* a makro F-míry u vybraných modelů a datových sad při trénování na trénovacích a validačních datech a vyhodnocení na testovacích datech.

C Struktura přiloženého DVD

- `readme.txt` – Textový soubor s detailním popisem struktury DVD a uživatelskou příručkou.
- `requirements.txt` – Textový soubor s výpisem knihoven v jazyce *Python* včetně jejich použitých verzí potřebných pro běh programu.
- `results` – Adresář se souborem ve formátu `xlsx`, který obsahuje podrobné výsledky experimentů – oproti textu této práce i míry *accuracy*, *weighted-precision* a *weighted-recall*, některé horší konfigurace modelů, výsledky při trénování pouze na trénovacích a testování na validačních i testovacích datech a dále hodnoty všech pěti běhů u modelů neuronových sítí včetně směrodatné odchylky. Jednotlivé nejlepší výsledky makro F-míry konfigurací modelů testovaných na validačních datech jsou zvýrazněny **tučně**. Soubor též popisuje parametry nutné ke spuštění daného experimentu. Listy `LSTM (old)` a `CNN (old)` obsahují výsledky experimentů s prvotním předzpracováním, které již nejdou zreplikovat.
- `src` – Adresář obsahující skripty ve formátu `py` a další podadresáře.
  - `data` – Adresář s použitými daty.
  - `dictionaries` – Adresář s použitými slovníky překladů.
  - `embeddings` – Adresář s vybranými předtrénovanými slovními vektory *fastText* a *word2vec*.
  - `tok_data` – Adresář s textovými soubory s tokenizovanými daty použitými pro trénování slovních vektorů *fastText* a *word2vec*. Tokeny jsou odděleny mezerami.
- `text` – Adresář obsahující text práce.
  - `BP_Jakub_Smid.pdf` – Samotný text práce ve formátu `pdf`.
  - `latex` – Adresář se zdrojovým kódem textu práce.
    - `obr` – Adresář s obrázky ve formátu `svg` použitými v textu bakalářské práce a jejich verze ve formátu `pdf` a `pdf_tex` nutné pro vložení do typografického systému LaTeX.